TRABALHO DE GRADUAÇÃO

MONITORAMENTO DE OCUPAÇÃO DE AMBIENTES USANDO RFID PASSIVO

André Abreu Rodrigues de Almeida

Brasília, Dezembro de 2019

ENGENHARIA
MECATRÔNICA
UNIVERSIDADE DE BRASÍLIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia
Curso de Graduação em Engenharia de Controle e Automação

TRABALHO DE GRADUAÇÃO

MONITORAMENTO DE OCUPAÇÃO DE AMBIENTES USANDO RFID PASSIVO

André Abreu Rodrigues de Almeida

Relatório submetido como requisito parcial de obtenção de grau de Engenheiro de Controle e Automação

Banca Examinadora

Prof. Dr. Adolfo Bauchspiess, ENE/UnB _____
Orientador

Prof. Lélio Ribeiro Soares Júnior, ENE/UnB _____
Examinador interno

Eng. Ney César de Melo Filho _____
Examinador externo

Brasília, Dezembro de 2019

FICHA CATALOGRÁFICA

ANDRÉ, ABREU RODRIGUES DE ALMEIDA Monitoramento de ocupação de ambientes utilizando RFID passivo [Distrito Federal] 2019. viii, 121p., 297 mm (FT/UnB, Engenheiro, Controle e Automação, 2019). Trabalho de Graduação – Universidade de Brasília. Faculdade de Tecnologia.	
1. RFID	2.Monitoramento
3. RSSI	4.Efeito Doppler
I. Mecatrônica/FT/UnB	II. Título (Série)

REFERÊNCIA BIBLIOGRÁFICA

ALMEIDA, ANDRÉ ABREU RODRIGUES DE, (2019). Monitoramento de ocupação de ambientes utilizando RFID passivo. Trabalho de Graduação em Engenharia de Controle e Automação, Publicação FT.TG-$n°$003, Faculdade de Tecnologia, Universidade de Brasília, Brasília, DF, 121p.

CESSÃO DE DIREITOS

AUTOR: André Abreu Rodrigues de Almeida

TÍTULO DO TRABALHO DE GRADUAÇÃO: Monitoramento de ocupação de ambientes utilizando RFID passivo.

GRAU: Engenheiro ANO: 2019

André Abreu Rodrigues de Almeida

UnB - Faculdade de Tecnologia, Campus Universitário Darcy Ribeiro, Brasília.

70910-900 Brasília – DF – Brasil.

Dedicatória

Dedico este trabalho à minha namorada linda!

André Abreu Rodrigues de Almeida

Agradecimentos

Agradeço primeiramente a Deus, por me guiar e proteger em minha jornada e em todos os meus dias.
Agradeço ao Professor Dr. Adolfo Bauchspiess por me oferecer a oportunidade de realizar este trabalho e me orientar em uma difícil tarefa de realizar este projeto em apenas um semestre.
Agradeço a banca examinadora, ao Professor Lélio Soares e ao Engenheiro Ney Filho, por me auxiliarem na minha formação acadêmica e profissional. Agradeço também aos meus colegas de trabalho na Johnson Controls, que me proporcionaram grandes aprendizados, crescimento profissional e boa companhia durante meus dois anos de estágio.
Agradeço especialmente à Maria Beatriz, minha namorada, por me proporcionar a melhor companhia do mundo, por ser a luz dos meus dias e me fazer sorrir sempre. Você me faz calmo em situações difíceis e feliz em todos os momentos. Obrigado por todo o seu amor e pelos incontáveis momentos especiais, e por toda a ajuda na elaboração deste trabalho!
Agradeço a David Halliday e Robert Resnick pelo livro de física que rendeu história!
Agradeço à minha família, ao meu pai João, minha irmã Beatriz, minha mãe Cynara, Minhas avós (Paulina e Bida), às minhas tias e tios, primas e primo. Vocês me moldaram para ser a pessoa que sou hoje.
Aos meus amigos, que me acompanharam durante toda a faculdade, me apoiando em todos os momentos e comemorando as minhas conquistas. Ao projeto Hulk: Renato, Lucas e Filipe, por serem sempre presentes. Ao Jessé, por todo o apoio, e pela companhia. Ao Hooper, Zago, Arthur, Caixeta, Geordana, Takashi, Redy, Luan e Marco Emílio e a todos que me acompanharam na UnB, aos meus amigos do Ciência sem Fronteiras e a todos os outros, muito obrigado!

André Abreu Rodrigues de Almeida

RESUMO

O presente trabalho utilizou a tecnologia de RFID passivo para o monitoramento e localização de transeuntes em ambientes fechados. A metodologia mostra seis estratégias diferentes para a identificação de mudança de ambiente por parte de um ocupante, utilizando as informações de RSSI e de efeito Doppler. Os resultados obtidos foram que a técnica que avalia a maior magnitude do sinal RSSI e a técnica que compara os tempos dos picos de potência obtiveram bons resultados. Já as técnicas que fazem a avaliação da média e mediana das últimas leituras de potência RSSI não se mostraram bons estimadores de posição. Por fim, os métodos que utilizaram efeito Doppler foram mais suscetíveis a falhas por perda de sinal. Nos apêndices deste trabalho foram dispostos o programa desenvolvido em C# e os registros das leituras e estimativas de cada método.

Palavras Chave: RFID, Monitoramento, RSSI, Efeito Doppler

ABSTRACT

The present work used passive RFID technology for monitoring and locating passers-by indoors. The methodology shows six different strategies for identifying an occupant's change of environments using the RSSI information and the Doppler effect. The results obtained were that the technique which evaluates the largest magnitude of the RSSI signal and the technique which compares the power peak times obtained good results. Moreover, the techniques that make the mean and median evaluation of the last RSSI power readings were not good position estimators. Finally, methods using the Doppler effect were more susceptible to signal loss failures. The appendices of this work present the program developed in C# and the logs of the readings and estimates of each method.

Keywords: RFID, Monitoring, RSSI, Doppler Effect

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

Símbolos Latinos

f	Frequência	[Hz]
T	Período	[s]
x	Coordenada - abscissa	[m]
y	Coordenada - ordenada	[m]
Z	Impedância elétrica	[Ω]
m	$Mili = 10^{-3}$	

Símbolos Gregos

Γ	Coeficiente de refração do sinal de retorno	
Ω	Ohm - unidade de impedância	
θ	Fase angular	[rad]
Δ	Variação	
π	Constante matemática - aproximadamente $3,14159265359$	
μ	$Micro = 10^{-6}$	

Grupos Adimensionais

k	Constante
m	Metro - unidade de distância
s	Segundo - unidade de tempo
dB	Decibel - unidade de proporção ou intensidade em escala logarítmica
Hz	Hertz - unidade de frequência
B	Quantidade de Bits no *chip* de uma TAG
C	Constante proporcional à quantidade de TAGs no campo de leitura de uma antena
M	Número de Miller - Número de ciclos por bit no processamento de uma TAG

Subscritos

CM	Combinada conjugada
SC	Curto-circuito
$Prop$	Propagação de ondas eletromagnéticas
o	deslocamento de fase (*offset*)
BS	retroespalhamento (*backscattering*)
A	Amostragem
OP	Operação

Siglas

DNS	Sistema de Nomes de Domínio - *Domain Name System*
EPC	Código Eletrônico do Produto - *Electronic Product Code*
EPC	Engenharia, aquisições e construção - *Engineering, Procurement and Construction*
ERP	Sistema de Gestão Empresarial - *Enterprise Resource Planning*
GPIO	Portas Programáveis de Entrada e Saída de Propósito Geral - *General Purpose Input Output*
GPS	Sistema de Posicionamento Global - *Global Positioning System*
HF	Alta Frequência - *High Frequency*
HVAC	Aquecimento, Ventilação e Ar-condicionado - *Heating, Ventilation and Air Conditioning*
ID	Identificação - *Identification*
IDE	Ambiente de Desenvolvimento Integrado - *Integrated Development Environment*
IoT	Internet das coisas - *Internet of Things*
IP	Protocolo de Internet - *Internet Protocol*
LARA	Laboratório de Automação e Robótica
LF	Baixa Frequência - *Low Frequency*
RAIN	Identificação por radiofrequência - *RAdio frequency IdentificatioN*
RF	Radiofrequência - *radiofrequency*
RFID	Identificação por Radiofrequência - *Radio Frequency Identification*
RSSI	Indicador de Intensidade do Sinal Recebido - *Received Signal Strength Indication*
UHF	Frequência Ultra Alta - *Ultra High frequency*
UnB	Universidade de Brasília

Capítulo 1

Introdução

O desenvolvimento tecnológico culminou na demanda por melhorias de serviços que, outrora, não eram capazes de integrar os fatores aliados à facilidade e ao bem-estar dos usuários. O bem-estar, dentro de edifícios, está associado diretamente ao conforto ambiental. Esse conceito é estudado nos ramos de arquitetura e construção civil e visa a garantia das condições que possibilitam a qualidade física e psicológica dos indivíduos dentro dos mais diversos ambientes e instalações.

1.1 Contextualização

O momento em que se elabora este projeto é uma época de grandes mudanças em quesitos de tecnologia. O entendimento do impacto ao meio ambiente, gerado pelo consumo energético em residências, estabelecimentos comerciais e fabris começa a ser compreendido por boa parte das pessoas, e principalmente o conhecimento da escalabilidade do impacto individual das pessoas de uma população.

Nas últimas décadas, a racionalização do consumo energético vem sendo alvo de diversos encontros técnicos da área de sustentabilidade, e de diversos acordos globais para a redução das emissões de carbono, geradas principalmente, devido à ação humana, especialmente na produção de energia.

Ao mesmo tempo em que o consumo é questionado, as exigências quanto à qualidade de vida, segurança no trabalho, ergonomia e conforto passaram a ser aspectos chave na vida e no dia-a-dia das pessoas. Seja no trabalho, ou no ambiente domiciliar, o conforto térmico é decisivo para o bem-estar das pessoas, assim como para a produtividade no ambiente de trabalho.

É um desafio alcançar a completude do conceito de conforto ambiental nas edificações. Nesse sentido, a automação é capaz de otimizar o uso de instrumentos cotidianos para se obter vantagens como: a facilitação de trabalhos, muitas vezes manuais, a segurança, o controle de acesso e a economia de energia, a exemplo da entrada de pessoas em uma sala e o acionamento do ar

condicionado, dentre outras vantagens.

1.1.1 Conforto ambiental

Conforto Ambiental é um conjunto de condições que possibilitam o bem-estar físico e psicológico dos indivíduos em diferentes locais, sendo que, o enfoque deste trabalho é principalmente em relação ao interior de instalações.

Esse conforto pode ser subdividido em categorias como: visual (abrangendo iluminação e estética visual), ergonômico (que tem muito destaque na parte de segurança do trabalho e com exemplos claros de esforços repetitivos, etc.), acústico (onde há limites em decibéis muito bem definidos) e térmico, que se configura como objeto desse trabalho. As normas vigentes que regem a construção e habitação de ambientes quanto ao aspecto de conforto ambiental e térmico são a ANSI/ASHRAE *Standard* 55-2017 [19] e as NBRs 15.575 [20] e 15.220 [21].

Nesse sentido, tecnologias de comunicação sem fio, como o RFID - que será melhor explicado no decorrer deste projeto – podem ser utilizadas para o melhoramento interno (*indoor*) de edifícios, proporcionando melhores condições para que seja possível atingir o conforto ambiental. A figura 1.1 demonstra a aplicação de tecnologias de controle e monitoramento para a aquisição de conforto interno.

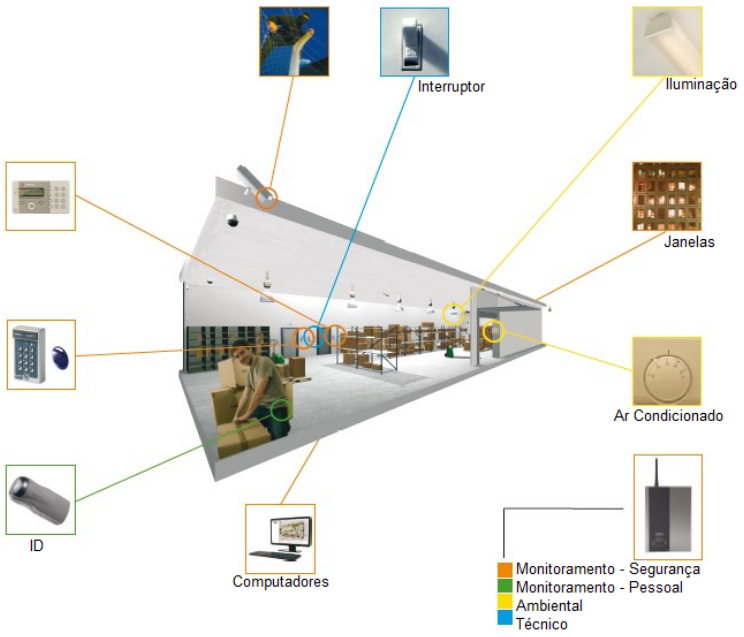


Figura 1.1: Tecnologias de controle e monitoramento *indoor* a fim de se atingir conforto ambiental. (Adaptado de 3DSecurity Systems [1])

1.1.2 Tecnologias de comunicação sem fio

O uso de RFID, Bluetooth, Zigbee pode auxiliar na automação e possibilitar a instalação de sistemas de condicionamento baseados na movimentação (entrada e saída) de pessoas. Segundo

Ahmed *et al.* [17], sistemas de comunicação sem fio (*wireless*) podem reduzir significativamente a frequência de erros humanos, otimizar a gestão e aumentar a acurácia da informação principalmente em relação à indoor network (rede interna). Chen *et al.* [5], retrata ainda, quão importante é a utilização de ferramentas como o RFID para a rastreabilidade e compra segura de alimentos. O uso de RFID pode ser empregado até no controle do fluxo de saída de roupas de uma loja.

Esses sistemas que funcionam à distância, sem fio, como o RFID, que é o sistema de identificação por radiofreqüência (RFID), que será melhor explicado no capítulo 2, permitem recuperar informações armazenadas de um objeto preso ou incorporado a bens, produtos ou seres vivos [22]. Dessa forma, o emprego de tecnologias sem fio pode proporcionar a melhor administração de indivíduos em situações rotineiras e também em situações de emergência.

1.2 Trabalhos anteriores

A elaboração deste projeto dá continuidade à uma sequência de trabalhos executados no Laboratório de Automação e Robótica da Universidade de Brasília (LARA - UnB). Em especial destacam-se o trabalho de Oliveira, Filipe e Rocha, Frederico [9] e o trabalho de Alves, Raissa e Chupel, Renata [23], este último a partir do qual dá-se continuidade. Este trabalho estima obter uma nova abordagem aos equipamentos de RFID disponíveis no laboratório, para obter a identificação da quantidade de pessoas em um ambiente, através de um novo algoritmo, e uma nova disposição física dos equipamentos no laboratório.

1.3 Proposta

A proposta deste trabalho consiste em utilizar a tecnologia RFID passivo para estimar a quantidade de pessoas que ocupam um determinado ambiente. O cenário vislumbrado é de um edifício de escritórios, sala de aula, anfiteatro ou auditório, oficina ou fábrica, ou qualquer ambiente cuja definição exata da quantidade de pessoas que ocupam tal ambiente seja relevante.

A motivação principal para a elaboração do trabalho foi a estimação do impacto na carga térmica causado pelas pessoas que transitam e ocupam um ambiente fechado. O princípio almejado é realizar o controle de equipamentos de ventilação, aquecimento e ar-condicionado (HVAC - em inglês *Heating, Ventilation and Air Conditioning*) de forma antecipada à variação das condições climáticas internas, em especial a temperatura e a umidade. Este propósito é considerado importante para sistemas de ar-condicionado simples e complexos, seja para manutenção do conforto térmico em um ambiente de trabalho, seja para minimizar a variação de temperatura em um ambiente onde as condições do ar são críticas, como uma sala de cirurgia ou laboratório, sala servidores ou *datacenter*.

Este propósito pode ser extrapolado para aplicações de controle de acesso e segurança, como por exemplo, a definição de alarmes de intrusão de áreas restritas, alarmes de quantidade máxima de ocupantes por sala, ou rastreamento da trajetória de visitantes em um ambiente. Utiliza-se como

exemplo um laboratório onde, para preservar as condições de limpeza, segurança e temperatura, limita-se o número de pessoas que pode transitar dentro deste ambiente por vez.

O principal objetivo deste trabalho é criar um espaço inteligente, que utiliza dados de tecnologias sem fio para aprimorar a qualidade de vida e de trabalho das pessoas, ao mesmo tempo em que proporciona economia de energia e segurança.

O objetivo específico do trabalho é desenvolver uma aplicação RAIN RFID para monitorar o trânsito de pessoas em um ambiente fechado e contabilizar a quantidade de pessoas presentes em cada subdivisão deste ambiente, em tempo real.

Esse trabalho apresenta, na fundamentação teórica (capítulo 2), o embasamento dos conceitos: tecnologias comumente utilizadas, RFID e seus componentes, redes de Petri, efeito Doppler e localização *indoor*.

À seguir, no capítulo 3 de Metodologias, são apresentados os materiais utilizados e as abordagens para se atingir o objetivo do trabalho, que é o monitoramento de pessoas em ambientes fechados. A seção de abordagem utilizada (seção 3.2) foi subdividida em: instalação física das leitoras e antenas, local de testes, elaboração do *software*, configuração das leitoras, transições e curvas de valores e os conceitos por trás dos seis métodos de monitoramento utilizados:

- Comparação do último valor de RSSI capturado
- Comparação dos tempos dos últimos picos das curvas de RSSI
- Comparação das médias das curvas de potência RSSI
- Comparação das medianas das curvas de potência RSSI
- Utilização do efeito Doppler como critério de travessia de um ambiente para outro
- Combinação dos métodos de tempos de picos de valores RSSI e transição por efeito Doppler

No capítulo 4 de Resultados foi apresentado um teste preliminar anterior ao desenvolvimento do *software*, que embasou a criação das seis abordagens citadas na metodologia. Foi apresentado o registro do teste do programa e a eficiência calculada para cada método.

Por fim, as conclusões são apresentadas no capítulo 5, com uma breve menção aos resultados de cada estratégia e uma visão geral do estudo realizado. O trabalho é finalizado apresentando as perspectivas futuras, tanto nos quesitos técnicos e de novas estratégias, como nos avanços práticos, para implementação em sistemas reais e integração com programas de automação existentes no mercado.

Capítulo 2

Fundamentação Teórica

2.1 Tecnologias comumente utilizadas

Existem diversos modos de estimar a ocupação de um certo ambiente por pessoas. Sensores de presença infravermelhos passivos já são amplamente utilizados comercialmente em aparelhos de ar-condicionado, especialmente modelos *split* para uso em pequenas salas.

Sensores passivos de presença, entretanto, não permitem um controle mais acurado da atuação do ar-condicionado. Eles auxiliam na economia de energia indicando a presença, ou não, de alguma pessoa ou animal no ambiente, mas sem indicar quantas pessoas estão presentes no local ou quais são suas características físicas. Além disso estão muito sujeitos a interferências, causando falsos positivos ou falsos negativos, devido ao modo de atuação passivo que apenas verifica a emissão de radiação infravermelha em comprimentos de onda próximos ao dos humanos.

Uma alternativa possível é o uso de RFID. Esta é uma tecnologia que se torna mais eficiente, acessível e popular a cada dia. O monitoramento posicional de objetos dentro de uma loja ou armazém tornou-se uma prática possível com RFID passivo, prática que é aplicada com soluções comerciais disponíveis no mercado.

2.2 RFID

A comunicação do tipo Identificação por Radiofrequência – *Radio Frequency Identification* (RFID) – é um tipo de comunicação sem fio por sinais de rádio. O RFID é uma alternativa de identificação útil em aplicações que exigem a leitura de grandes quantidades de dados [24].

A composição geral do RFID é feita por uma *TAG* (etiqueta ou transpônder) e uma leitora (*reader*) que são incorporadas com um transmissor e um receptor [2]. A leitora (ou interrogadora) possui uma antena para comunicação com as TAGs e, na maioria dos casos, uma saída para comunicação com um computador ou servidor. Alguns autores adicionam, como parte de um sistema RFID, uma interface homem-máquina (IHM) e *softwares* de ERP (Sistema de gestão empresarial, em inglês - *Enterprise Resource Planning*) para EPCs (Engenharia, aquisições e

construção, em inglês - *Engineering, Procurement and Construction*), ou ainda, programas e dispositivos de computação em nuvem. Isso pode ser observado na figura 2.1.

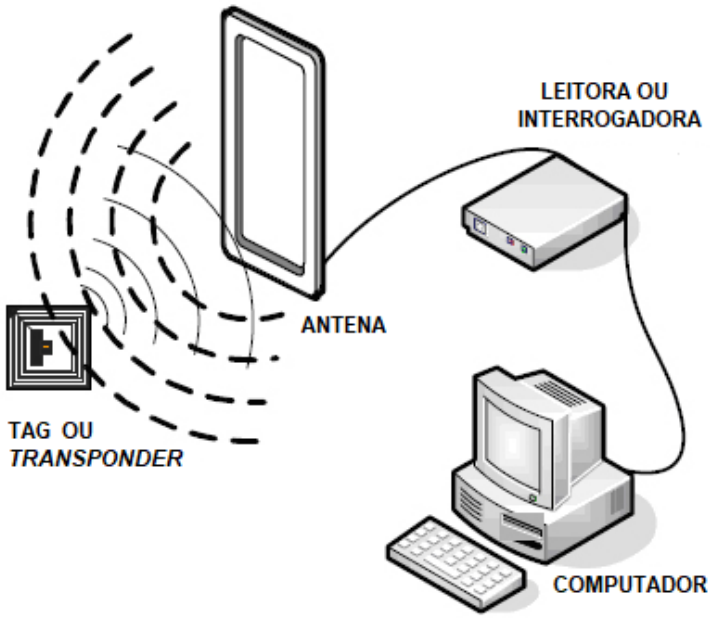


Figura 2.1: Composição básica de um sistema RFID. (Adaptado de EPC-RFID [2])

As primeiras aplicações comerciais de RFID, como artigo eletrônico de vigilância (EAS - *Electronic Article Surveillance*), aconteceram no final da década de 1960, desenvolvidas por empresas como: Kongo, Sensormatic (by Johnson Controls) e Checkpoint [4].

De acordo com Chawla e Ha [4], o sistema RFID consiste em leitoras e TAGS. A leitora tem como função se comunicar na faixa de alcance sem fio e coletar informações sobre os objetos aos quais as TAGS estão anexadas (ex.: entrada e saída de pessoas de um estabelecimento).

O funcionamento simplificado do RFID passivo pode ser observado na figura 2.2. Um gerador de sinais de radiofrequência (RF - *radiofrequency*) emite um sinal à um transpônder com impedância variável e um codificador de identidade. O sinal emitido excita o oscilador presente na TAG,que por sua vez reflete o sinal modificado com as informações de identidade da TAG. O sinal de resposta é capturado por uma antena, amplificado e decodificado. A informação de identidade da TAG pode então ser processada, apresentada ou armazenada.

A figura 2.3 mostra os elementos internos dos dispositivos de um sistema RFID. Nela é possível observar o trajeto de dados da energização da bobina da TAG e envio de informação através da variação de potência do campo magnético. A energização alimenta o circuito da TAG que é composto de carga, codificador RF e chip de controle, memória e criptografia. O circuito excitado da TAG reflete o sinal codificado e é capturado pela bobina da antena da leitora, que interpreta o sinal e retransmite a informação em uma interface de rede para um computador.

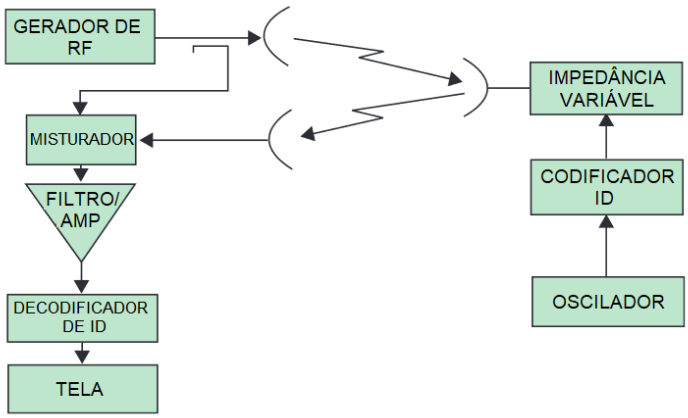


Figura 2.2: Diagrama de rede de uma comunicação RFID. (Adaptado de Landt, Jeremy [3])

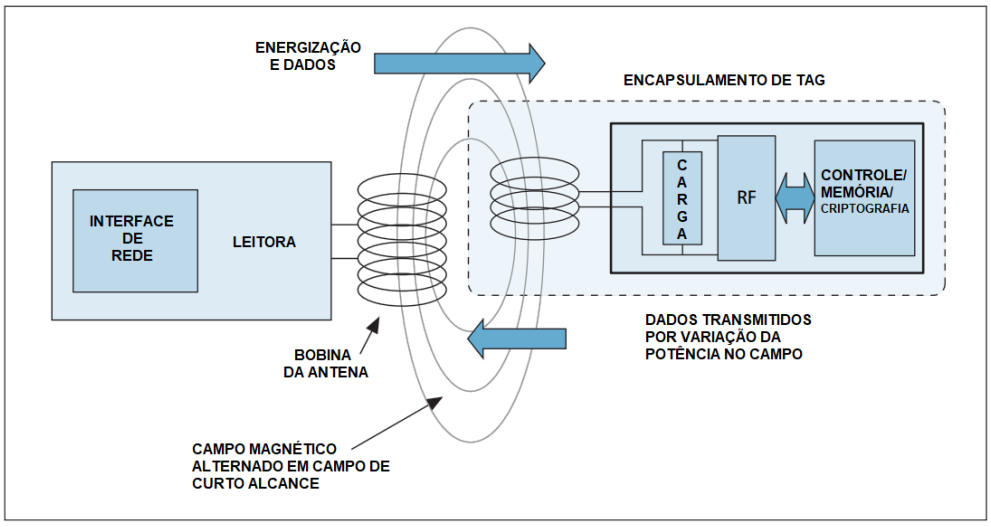


Figura 2.3: Estrutura interna básica dos elementos de um sistema RFID. (Adaptado de Chawla e Ha [4])

2.2.1 *Middleware*

Chen et al. [5] define RFID como o conjunto de TAGs, leitoras, *middleware* e sistema de aplicação, como pode ser visto na figura 2.4.

Middleware é o *software* que liga o Sistema Operacional às aplicações. É comum definir um *middleware* como um "encanamento"que leva dados de um ponto a outro. [25]

No caso do RFID, o *middleware* é um *software* que interpreta e organiza os dados enviados pelas leitoras. Ele realiza a filtragem e agregação da informação, a coordenação da leitura, o roteamento de dados e integração e o gerenciamento do processo. Ao final do processamento, ele disponibiliza os dados para aplicações que podem ser, por exemplo, ERPs (em inglês: *Enterprise Resource Planning* - Planejamento de Recursos Empresariais), sistemas de informação ou um *software* de localização de pessoas em ambientes fechados.

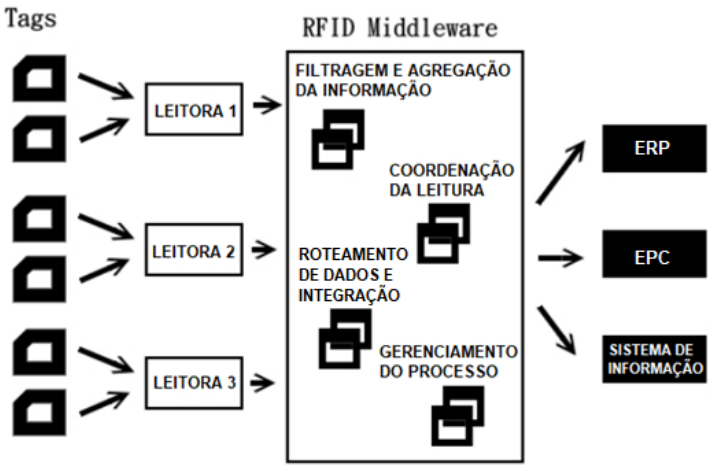


Figura 2.4: Topologia de comunicação do *middleware* RFID (Adaptado de Chen *et al.* [5])

2.2.2 RAIN RFID e IoT

O padrão RAIN (acrônimo para *RAdio frequency IdentificatioN* - Identificação por radiofrequência) é a união de diferentes tipos de *hardware* e *software* que possuem o propósito de ligar o meio RFID UHF com a computação em nuvem. É padronizada pela organização global de mesmo nome (RAIN) com o objetivo de popularizar o uso de RFID. [26]

RAIN RFID foi desenvolvido principalmente para aplicações de internet das coisas (IoT - *Internet of things* em inglês). IoT é, segundo Gubbi, *et al.* [27]:

> "Interconexão de dispositivos de detecção e atuação, oferecendo a capacidade de compartilhar informações entre plataformas por meio de uma estrutura unificada, desenvolvendo uma imagem operacional comum para permitir aplicativos inovadores. Isso é conseguido através da detecção onipresente e contínua, análise de dados e representação de informações com a computação em nuvem como a estrutura unificadora."[27]

2.2.3 *TAGs*

TAGs, também conhecidas como transpônderes, são etiquetas ou chips microprocessadores de identificação. Esses dispositivos recebem sinal de rádio e transmitem, automaticamente, um sinal diferente. Cada TAG possui um número serial de identificação que a difere de outras. [4]. Esses objetos podem armazenar informações como *serial number* (número serial), *model number* (número do modelo), bem como outras características como: data, cor, tamanho e preço. A TAG RFID é classificada de acordo com sua memória, tipo de comunicação e uso de energia, como é possível visualizar na tabela 2.1 [17].

Tabela 2.1: Classificação de TAGs RFID (Adaptado de [17])

Origem da alimentação elétrica	
Passiva	Alimentada pela leitora através do próprio sinal de radiofrequência. Também chamada de 'reflexiva', 'alimentada por feixe' ou de 'retroespalhamento' (*backscattering*).
Semi-Passiva	Utiliza uma bateria para manter a memória na etiqueta ou alimentar os componentes eletrônicos que permitem que a etiqueta module o sinal refletido.
Ativa	Alimentada por uma bateria interna. Possui maior alcance de leitura e é mais cara que as TAGs passivas. As baterias devem ser substituídas periodicamente.
Tipo de Memória	
Somente leitura	A memória é programada na fábrica e não pode ser modificada após sua fabricação. Geralmente armazena apenas 96 bits de informação.
Leitura-escrita	A leitora RFID pode ler e escrever na memória desta TAG. É possível armazenar uma quantidade maior de dados (de 32 kb a 128 kb). É mais cara que os chips do tipo somente leitura.
Tipo de Comunicação entre a TAG e a Leitora	
Indução	Acoplamento eletromagnético ou indutivo de proximidade. Utilizada em Bandas de frequência LF e HF.
Propagação	Propagação de ondas eletromagnéticas - campo distante. Opera nas faixas de frequência UHF e microondas.

É possível classificar as TAGs em três tipos, como pode ser visto na tabela 2.1: ativas, passivas e semi-passivas [4]. Existem vários tipos de TAGS ativas (alimentadas por baterias) e passivas, em várias faixas de frequência, presentes no mercado [24]. A Figura 2.5 mostra exemplos de TAGS ativas e passivas.

Segundo Rao [24], TAGs ativas, como as apresentadas na figura 2.5A, tem como vantagem o alcance de distância de leitura maior do que as TAGs passivas, devido à sua construção com alimentação de energia própria. Como desvantagem, elas podem ser bem mais caras e exigem o uso de baterias [24]. Uma etiqueta RFID passiva, figura 2.5B, por sua vez, usará a energia das ondas de rádio do interrogador para retransmitir suas informações armazenadas de volta a ele [2].

As TAGs passivas podem possuir diversas construções de memória diferentes. O tipo mais simples é transpônder de 1-bit, muito utilizado em TAGs EAS (Artigo Eletrônico de segurança - *Electronic Article Surveillance* em inglês). Ele possui um bit de memória apenas. O seu funcionamento é simples, a TAG responde ao interrogador um sinal de presença naquele ambiente,

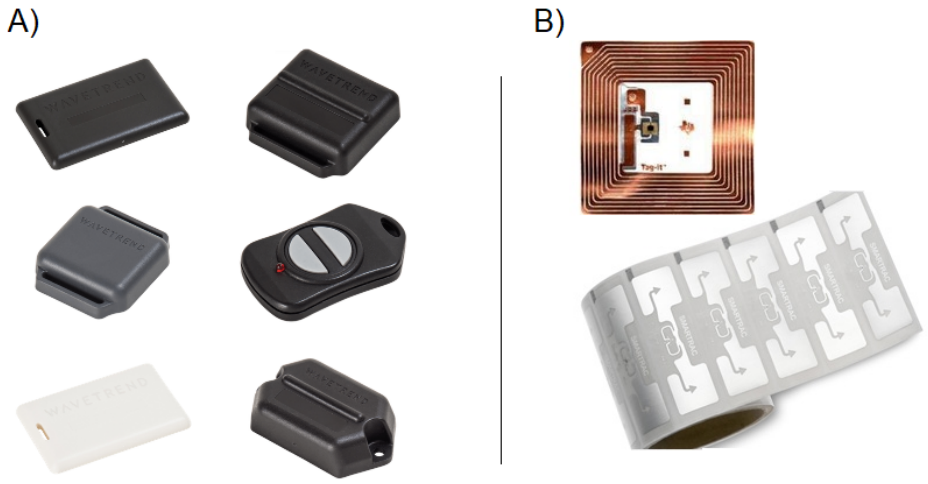


Figura 2.5: A) Exemplos de TAGs ativas [6] - B) Exemplos de TAGs passivas [7] [8]

armazenando neste bit a informação de resposta, *true* ou *false*. [28]

Já os transpônderes de n-bits podem possuir de poucos bits até 128 kb a depender do sistema. Eles se dividem em *full-duplex*, *half-duplex* e sequenciais. A diferença entre eles é o protocolo de comunicação entre a TAG e o interrogador. [28]

- *Half-duplex*
  - O interrogador alimenta o transpônder continuamente
  - O interrogador e o transpônder se alternam para emitir dados de comunicação, com o transpônder utilizando frequências harmônicas para responder ao interrogador
- *Full-duplex*
  - O interrogador alimenta o transpônder continuamente
  - O interrogador e o transpônder emitem dados de comunicação simultaneamente, com o transpônder utilizando frequências subarmônicas e anarmônicas para responder ao interrogador
- Sequencial
  - O interrogador alimenta o transpônder somente durante a sua etapa de transmissão de dados;
  - A transferência de dados do transpônder para a leitora ocorre somente durante curtos períodos de tempo, como uma resposta ao pulso de alimentação.

2.2.4 Antenas

A antena é o equipamento de um sistema RFID capaz de perceber o sinal de uma TAG dentro de um determinado raio de cobertura. Elas se conectam às leitoras e são responsáveis por propagar os sinais enviados pela leitora no meio, energizar as etiquetas e escutar o sinal de resposta.

Antenas são dispositivos usados na transferência e captação de ondas eletromagnéticas em meios ilimitados no espaço. São projetadas em bandas de frequência de operação. Dessa forma, um sinal ou é atenuado ou é rejeitado por completo. [29]

2.2.5 Efeitos causados por materiais e frequência de operação

É amplamente conhecido na indústria do RFID e de outras tecnologias que trabalham com RF (radiofrequência - *radiofrequency* em inglês) que os materiais presentes no ambiente onde se propagam os sinais transmitidos interferem com esse sinal.

Estudos com materiais mostram que podem existir nove tipos de perdas no sinal, como mostra Sanghera em seu livro[18]:

- **Absorção:** Energia do sinal absorvida pelo meio;
- **Atenuação:** Diminuição da amplitude do sinal devido à absorção e dispersão;
- **Efeitos Dielétricos:** Capacidade do meio de reter carga. Gera atraso do sinal;
- **Difração:** "Dobra"de um sinal de radiofrequência ao atingir uma quina ou pequeno orifício;
- **Perda de espaço livre:** Diminuição da densidade do sinal devido à característica de propagação do sinal para todos os lados;
- **Interferência:** Ocorre no encontro de duas ondas de radiofrequência, pode ser construtiva ou destrutiva;
- **Reflexão:** Retorno do sinal em um ângulo isóceles ao de incidência em uma superfície plana. Geralmente ocorre no chão, teto e paredes;
- **Refração:** mudança de direção de trajetória do sinal, entretanto permeando o meio causador da refração;
- **Espalhamento:** Absorção e re-emissão do sinal, em direções aleatórias.

Sanghera mostra ainda uma tabela contendo a influência de cada material no sinal de radiofrequência [18] que é adaptada e apresentada a seguir (Tabela 2.2).

Tabela 2.2: Efeito dos materiais nos sinais de RFID (Adaptado de [18])

Material	Efeito no sinal RF
Papelão	Absorção
Líquido condutivo	Absorção
Vidro	Atenuação
Conjunto de latas	Reflexão em múltiplas direções
Corpo humano ou animal	Absorção, dessintonização e reflexão
Metal	Reflexão
Plástico	Dessintonização devido ao efeito dielétrico

Um estudo conduzido por Fletcher [30] questiona, contudo, os efeitos dos sinais RF nos meios aquosos, e portanto, os efeitos no corpo humano e de animais por consequência. Ele afirma que, ao contrário do que se acredita, os líquidos interferem nas leituras dos sinais de RFID majoritariamente devido aos efeitos de reflexão e refração, enquanto o consenso é de que meios aquosos absorviam os sinais, principalmente.

De uma forma ou de outra, as leituras RFID obtidas são de baixa qualidade quando os transpônderes ou as antenas estão próximas de meios aquosos ou pessoas e animais. Esse é o principal motivo de o monitoramento de localização por RFID, ou outras tecnologias RF, não possuir muita aderência para o propósito de monitorar pessoas enquanto a tecnologia avança a passos largos no monitoramento de produtos e aplicações de *IoT*, visto que estes últimos são menos suscetíveis à perda de sinal.

O livro de Thornton e Sanghera [10] mostra a influência causada pela escolha da frequência de funcionamento do sistema. Os protocolos RFID preveem o funcionamento da tecnologia em quatro faixas de frequência:

- Baixa frequência (LF - *Low Frequency*)
  - opera a 125 kHz ou 134 kHz para dispositivos RFID
  - Baixa distorção devido a água
  - Curto alcance (até 10 cm)
  - Utilizado para identificação de pessoas e animais à curta distância
- Alta frequência (HF - *High Frequency*)
  - opera a 13.56 MHz para dispositivos RFID
  - alcance de até 3m
  - sinal RF com comprimento de onda pequeno / baixa capacidade de penetração em materiais
  - mais opções de velocidade de leitura em relação à baixa frequência
  - utilizado em controle de acesso e rastreio de itens como bagagens
- UHF (em inglês *Ultra High Frequency* - Frequência Ultra Alta em tradução livre para o português)
  - Diferentes frequências em diferentes países
    - Faixa de 860-960MHz
    - 902-928MHz no Brasil (geralmente 915MHz)
  - alta distorção devido a água
  - a frequência de operação depende do país onde o equipamento é homologado
  - maior alcance que LF e HF
  - usado para gerenciamento de estoque e armazéns

- Micro-ondas
  - Faixa de 2,44-5,80GHz para dispositivos RFID
  - Alta velocidade de leitura
  - Longo alcance
  - baixo desempenho próximo a água
  - utilizado para identificação de veículos e *supply chain* (em português, "cadeia de suprimentos"ou "cadeia de logística")

A imagem 2.6 de Oliveira e Rocha [9], traduzida da original no livro de Thornton e Sanghera [10], simplifica os conceitos do impacto da frequência de operação no sinal RFID.

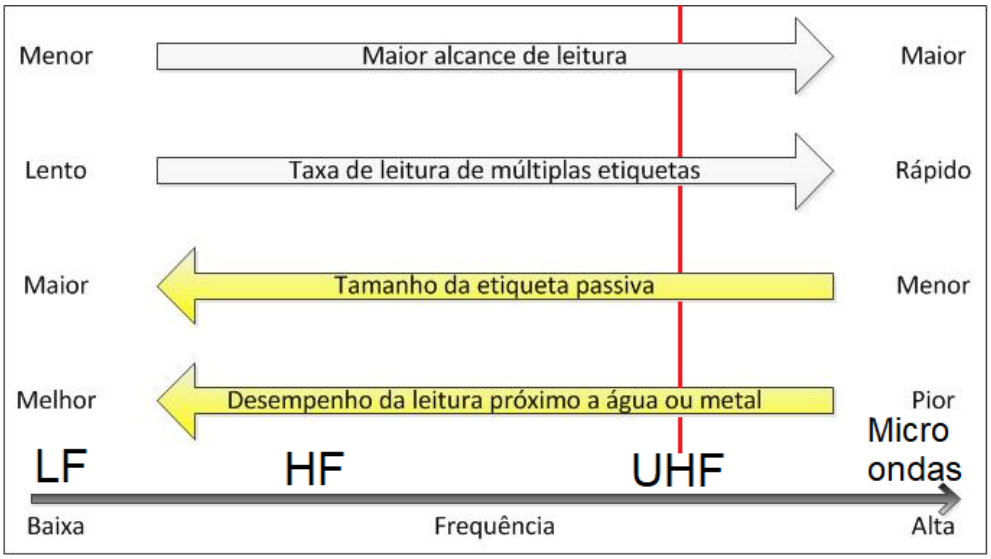


Figura 2.6: Impacto da frequência de operação nas leituras RFID. (Adaptado de [9] e [10])

A caracterização do sistema e das etiquetas passivas utilizados pode ser caracterizada na porção mais à direita do gráfico da figura 2.6, por se tratar de um sistema UFH, o que indica que as TAGs não necessitam de um tamanho exageradamente grande, e que o alcance do sistema é maior.

O gráfico da figura 2.6 pode ser analisado alocando os sistemas LF à esquerda, onde o alcance é menor e o desempenho próximo à água e metais é melhor, progredindo para a direita na ordem: HF, UHF e micro ondas. A cada progressão de banda de frequência, o alcance do sistema é aprimorado à custa do desempenho quando próximo de materiais problemáticos para sinais RF.

2.2.6 RSSI

O Indicador de Intensidade do Sinal Recebido (*Received Signal Strength Indication*), em poucas palavras, é a medida de quão eficiente um dispositivo pode ser na detecção de um sinal/ruído.

Segundo Buffi [11], existem duas impedâncias intrínsecas à cada TAG que geralmente correspondem à carga de impedância correspondente (Z_{CM}) e uma carga de impedância de curto-circuito

(Z_{SC}). Durante a comunicação entre a leitora RFID e a TAG alterna-se entre as duas impedâncias, como mostra a figura 2.7.

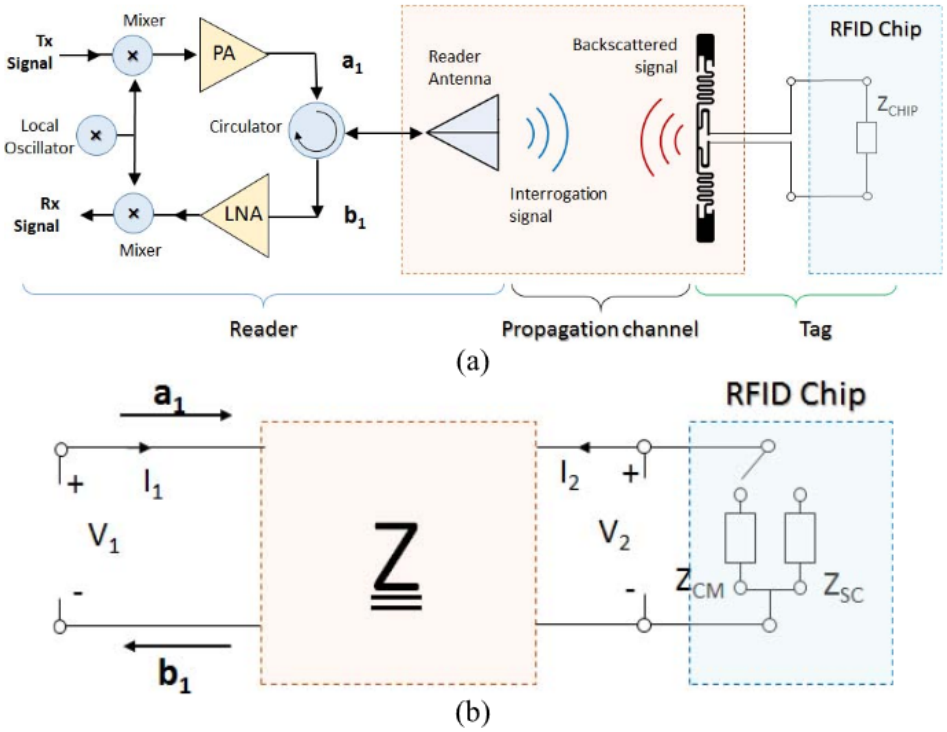


Figura 2.7: (a) Representação esquemática de uma TAG equipada com uma antena, um chip RFID e um leitor, este último constituído por uma antena, um circulador, um amplificador de potência para o sinal transmitido e amplificador de baixo ruído para o sinal recebido. (b) Matriz de impedância equivalente que representa a antena do leitor, a antena do TAG e o canal de propagação entre elas. Para gerar um *backscattering* modulado, a antena da TAG é conectada a uma carga de impedância combinada conjugada (Z_{CM}) ou a um curto-circuito (Z_{SC}). [11]

O sinal RSSI é calculado a partir do módulo da diferença dos coeficientes de refração Γ as amplitudes do sinal enviado e do sinal recebido, como mostra a equação 2.1.

$$RSSI = k|\Gamma_{CM}(x,y) - \Gamma_{SC}(x,y)|^2 \tag{2.1}$$

Na equação 2.1 k representa uma constante, x e y representam as coordenadas da TAG em relação à leitora, e Γ_{CM} e Γ_{SC} representam os coeficientes de refração do sinal de retorno. O coeficiente Γ é proporcional à relação entre as amplitudes do sinal enviado pela leitora e pelo sinal de retorno da TAG lido pela antena, de acordo com a equação 2.2.

$$\Gamma = \frac{b_1}{a_1} \tag{2.2}$$

Na equação 2.2 a_1 é a amplitude do sinal enviado pela leitora e b_1 é a amplitude do sinal de

resposta capturado pela antena.

As equações 2.1 e 2.2 discutidas nesta seção são apresentadas para informar o leitor da origem e dos cálculos feitos internamente pela leitora Impinj *Speedway* R420, modelo utilizado neste trabalho e apresentado na seção 3.1.1.1. Estes cálculos são feitos diretamente em *hardware* e tratados no *middleware* da leitora e, portanto, não se executam tais cálculos neste trabalho. As informações de RSSI são adquiridas diretamente da leitora para uso do programa criado neste trabalho.

2.3 Rede de Petri

A Rede de Petri consiste em uma ferramenta gráfica e matemática que possui inúmeras aplicações. Segundo Francês [31], a representação gráfica básica de uma rede de Petri compreende dois componentes principais: um ativo (transição) - que graficamente é uma barra - e um passivo (lugar), graficamente representado pelo círculo.

Algumas de suas aplicações são a de avaliar desempenhos, possibilitar a concepção de software em tempo real e/ou distribuído, avaliar o gerenciamento bases de dados, interfaces homem-máquina, atuar em sistemas de informação, controle, logística e transporte, além de outras inúmeras funcionalidades. As vantagens do uso podem ser descritas brevemente a seguir: [32]

- descrição de ordem parcial em eventos, possibilitando avaliar a flexibilidade;
- representação explícita dos estados;
- facilitação nos níveis da estrutura hierárquica do controle;
- utilização de uma única família de ferramentas;
- descrição precisa e formal das sincronizações, garantindo a segurança do funcionamento;

2.4 Efeito Doppler

O Efeito Doppler ocorre quando ondas são emitidas ou refletidas por um objeto em movimento. É um fenômeno físico importante para a análise da detecção de sinais em movimento.

As leitoras de RFID geralmente possuem capacidade de calcular o desvio de fase do sinal de resposta de *backscatter* das TAGs, de acordo com a equação 2.3. A partir dessa informação, é possível obter o valor da frequência Dopler causada pelo deslocamento da TAG. [33] [12]

$$\theta = \theta_{Prop} + \theta_o + \theta_{BS} \tag{2.3}$$

Na equação 2.3 θ representa a fase do sinal recebido da TAG, θ_{Prop} é a fase acumulada devido à propagação de ondas eletromagnéticas, θ_o é o deslocamento de fase (*offset*) nos cabos e antenas e θ_{BS} é a fase de retroespalhamento (*backscatter*) da modulação de TAG. [33] [12]

As leitoras RFID utilizam a fase do sinal de retorno θ para calcular a frequência Doppler através da equação 2.4 a seguir.[12]

$$f_D = \frac{\Delta\theta}{4\pi\Delta T} \tag{2.4}$$

Na equação 2.4 f_D é a frequência Doppler. Os valores de $\Delta\theta$ e ΔT podem ser obtidos com as equações 2.5 e 2.6, respectivamente.[12]

$$\Delta\theta = \theta_2 - \theta_1 \tag{2.5}$$

$$\Delta t = t_2 - t_1 \tag{2.6}$$

Na equação 2.5, θ_1 e θ_2 são as leituras de fase da TAG nas posições iniciais e finais de leitura, calculados a partir da equação 2.3, como pode ser visualizado na figura 2.8. Na equação 2.6, t_1 e t_2 correspondem aos tempos das leituras de θ_1 e θ_2 da equação 2.5. [12]

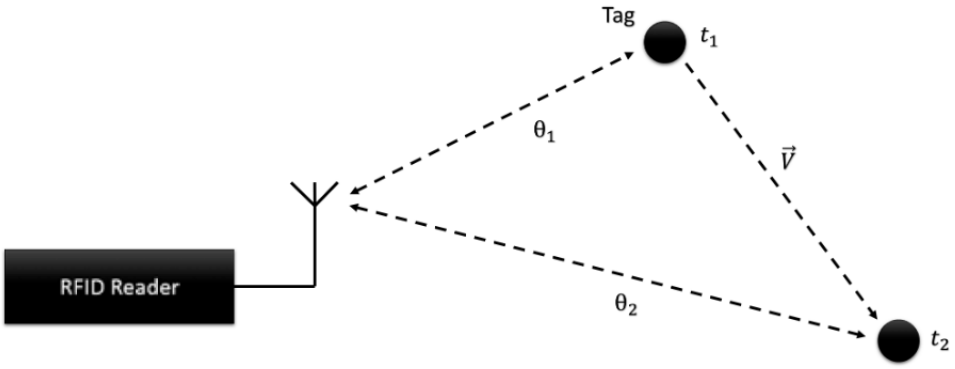


Figura 2.8: Frequência Doppler calculada a partir da fase do sinal de retorno da TAG [12]

A figura 2.8 mostra uma TAG em duas posições espaciais diferentes, em tempos t_1 e t_2 diferentes. Em cada posição o sinal de leitura retorna à antena com uma fase θ diferente.

Assim como o valor de RSSI, o valor de frequência de efeito Doppler é calculado internamente em *hardware* e *middleware* da leitora Impinj *Speedway* R420. Não será calculada a frequência Doppler neste trabalho, sendo que as leituras realizadas pelo equipamento utilizado já informam, juntamente às outras informações da TAG (número EPD e valor RSSI), o valor de frequência Doppler.

2.5 Localização *indoor*

O rastreio e localização de pessoas em ambientes fechados é algo fortemente desejado desde o surgimento das tecnologias de localização por sinais de radiofrequência. É imaginado um futuro

onde equipes de segurança saibam exatamente onde resgatar uma pessoa com dificuldades, dentro de um edifício; sistemas de ar-condicionado dimensionam automaticamente as suas potências baseados na ocupação do local aumentando a eficiência energética; mapas de ambientes fechados públicos que indicam o caminho até o destino final interno.

A localização de pessoas e objetos já está muito avançada para ambientes externos, com exemplos de muito sucesso como o sistema de posicionamento global - GPS (*Global Positioning System* em inglês) e seus similares (Galileo, Glonass, Compass), e também do Cell-ID - tecnologia de localização de dispositivos GSM/WCDMA/CDMA por triangulação de torres de celular. Contudo, a localização em ambientes fechados ainda é muito restrita, principalmente devido às restrições de comunicação por sinais RF em ambientes passíveis de obstrução por objetos e fontes de reflexão e demais tipos de interferência. Problemas como a atenuação e dispersão do sinal, sinais duplos ou múltiplos, e a grande quantidade de dispositivos comunicando-se em um pequeno espaço em algumas aplicações (o que gera conflito de diversas tentativas de estabelecimento de comunicação ao mesmo tempo), dificultam o desenvolvimento de uma tecnologia definitiva para esta aplicação.

Para desviar dos empecilho, soluções específicas para diversas aplicações são criadas. A indústria larga na frente, com sistemas de rastreio e posicionamento de produtos e equipamentos, como em armazéns e linhas de produção. O comércio também já possui soluções para prevenção de perdas e roubos, e começa a aplicar os conceitos de rastreamento de produtos de armazéns para identificar roupas e sapatos em estantes, por exemplo. A área de soluções para cidades pesquisa tecnologias IoT.

A área de localização de pessoas em ambientes fechados possui pesquisas com diversas tecnologias, como WiFi, Bluetooth, GSM, Radiação Infravermelha, processamento de imagens, RFID ativo e passivo, entre outras. Todas possuem vantagens e desvantagens.

A grande diferença entre as técnicas de rastreamento do RFID ativo e passivo é que, geralmente, as propostas com RFID ativo utilizam algum tipo de triangulação de sinais, com um transpônder e diversas leitoras [34] [35] [36], devido às características de funcionamento desse sistema. As estratégias com RFID passivo costumam necessitar de pontos de interrogação e controle de acesso para funcionar, devido ao curto alcance da maioria dos sistemas. Entretanto, algumas aplicações funcionais de localização cartesiana recentes utilizando frequências mais altas como UHF e Microondas [37] mostram que é possível minimizar as dificuldades de implementar sistemas de localização com RFID passivo.

Algumas implementações, como a de Tesoriero *et al.* [38], inovam ao misturar sensores ativos e passivos. Sua pesquisa apresenta o ambiente de um museu, onde os visitantes carregam PDAs (Assistentes Pessoais Digitais - em inglês *Personal Digital Assistants* que são leitoras RFID passivo, e as TAGs marcam os locais e as obras, invertendo o papel padrão dos elementos, e criando um mapa do museu a partir do posicionamento das TAGs.

Utilizando os recursos disponíveis na UnB e no LARA, este trabalho propõe uma solução própria, que será explanada no capítulo 3 - Metodologia.

2.6 Cálculo de taxa de amostragem

A fabricante das leitoras relata a relação do modo de leitura definido na figura 3.17 com a frequência de identificação dos cartões. Segundo a Impinj [13], o modo *"DenseReaderM8"* leva oito ciclos para processar cada bit de leitura, como pode ser visto na figura 2.9. Considerando que a leitora funcione à taxa de 900MHz, arredondando os valores reais de frequência de operação, realizam-se os cálculos da equação 2.7 para calcular o menor tempo possível de período de leitura.

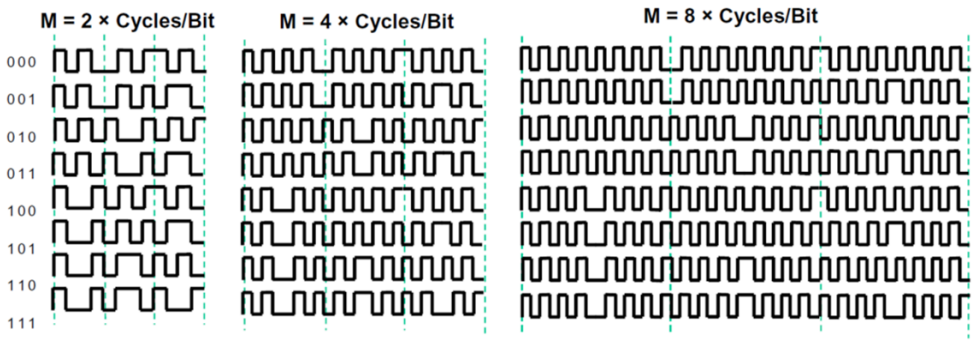


Figura 2.9: Modulação de modos de leitura [13]

$$T_A = \frac{1}{f_{op}} * M * B * C \tag{2.7}$$

Na equação 2.7, a variável T_A representa o período de amostragem mínimo, f_{op} é a frequência de operação, M é o modo de leitura (quantos ciclos por bit são necessários), M é o número de Miller [13], que registra o numero de ciclos por Bit lido, B é o número de Bits total do chip da TAG e C é uma constante que relaciona o atraso na leitura provocada por existirem múltiplas TAGs no campo de cobertura da antena. Realizando os cálculos para o caso específico deste trabalho, obtém-se o valor calculado na equação 3.1.

Os valores de taxa de amostragem serão calculados na seção 3.3.

Capítulo 3

Metodologia

Este capítulo apresenta o processo de desenvolvimento do programa de monitoramento de ocupação de ambientes. A descrição dos materiais utilizados, da instalação física e a criação de estratégias de localização de pessoas são os destaques do texto.

3.1 Materiais

A execução desse trabalho contou com os materiais de excelente qualidade disponíveis no laboratório e adquiridos para esta aplicação específica, que foram apresentadas na seção 3.1.1, assim como as ferramentas computacionais descritas na seção 3.1.2.

3.1.1 *Hardware*

Nesta parte do texto foram apresentadas as especificações, capacidades e limitações de equipamentos RFID - mais especificamente, a leitora, antena e TAGs utilizadas.

3.1.1.1 Leitora RFID Impinj Speedway R420

A Speedway R420 da fabricante Impinj é uma pequena leitora estacionária de *tags* de RFID passivo. Ela é capaz de operar na faixa de frequência UHF (860-960 MHz), sendo que o modelo específico licenciado para o Brasil permite operações entre 902,5 e 907 MHz. Pode emitir sinais de até 32,25 dBm de potência na transmissão e é capaz de receber um sinal com a sensibilidade mínima de -80 dBm e perda de retorno de 10 dB [39] [14] [9]. A figura 3.1 mostra uma foto da leitora.

A leitora é capaz de registrar até 1100 etiquetas por segundo. Ela possui portas suficientes para o acoplamento de até 4 antenas, expansível até 32 antenas utilizando *Hubs* específicos para esta aplicação. Dois protocolos são utilizados para a interface por ar: *GS1/EPCglobal UHF Gen2 (ISO 18000-6C)* e *RAIN RFID* [39][14]. As portas de conexão das antenas podem ser vistas na

Figura 3.1: Leitora Impinj Speedway R420 [14]

figura 3.2.

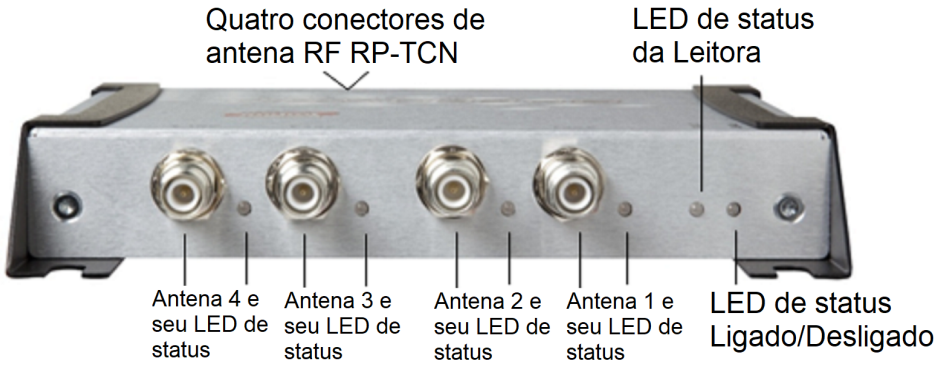


Figura 3.2: Leitora Impinj Speedway R420 - vista frontal - Adaptado de [14].

Existem diversas portas na parte traseira da leitora Speedway R420. Uma delas é uma porta *10/100 BaseT Ethernet* para comunicação em TCP/IP. A comunicação por TCP/IP é ideal para o uso comum, utilizando os softwares comercializados pela Impinj ou por programas feitos utilizando o pacote *Impinj OctaneSDK*. Além dessa porta, a comunicação pode ser feita por USB ou por comunicação serial RS-232 em um conector RJ45 para acesso ao console embarcado na leitora. As portas seriais são ideais para utilização dos pacotes *Impinj LTK* de programação em baixo nível, utilizando o padrão *Low Level Reader Protocol* (LLRP). Baixo nível, neste caso, implica capacidade de alterar o protocolo de operação, o modo de marcação de tempo e dos parâmetros de comando por ar[40][14]. A visão traseira da leitora é apresentada na figura 3.3.

Existe ainda uma porta *GPIO DE-15* (*General Purpose Input-Output DE-15* - Porta de uso de próstio geral para entrada e saída de dados)[14]. Ela pode ser utilizada para a configuração de

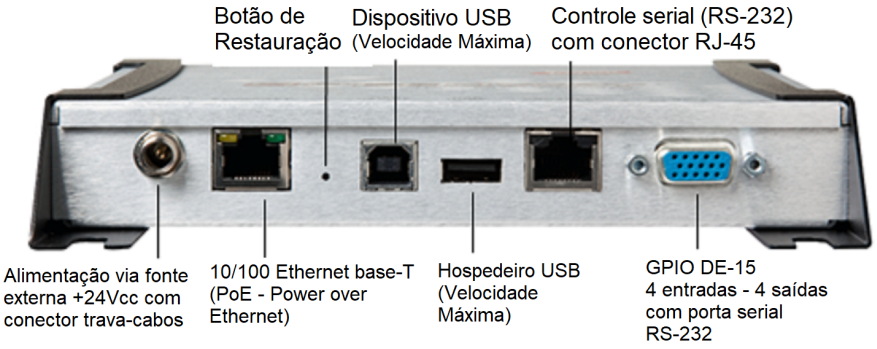


Figura 3.3: Leitora Impinj Speedway R420 - vista traseira - Foto obtida no manual de instalação e operação da leitora - Adaptado de [14].

comandos e gatilhos de leitura.

A alimentação elétrica da leitora pode ser feita utilizando o padrão *PoE* (*Power over Ethernet*) [14], ou seja, sem a necessidade de fonte externa, apenas através do cabo Ethernet, ou então pode ser feita com uma fonte 24 volts de corrente contínua [41]. A fonte pode prover uma confiabilidade maior para operação com potências maiores, próximas ao limite de 32,25 dBm, enquanto a alimentação *PoE* permite maior flexibilidade na instalação do produto, principalmente em relação à passagem de cabos e instalação de tomadas.

3.1.1.2 Antena RFID Impinj Threshold

A Threshold, também da fabricante Impinj, é uma antena RAIN RFID específica para detectar cruzamento de barreiras e fronteiras. O seu corpo alongado, com polarização linear paralela ao menor eixo, permite uma ampla cobertura, e a possibilidade de conectar múltiplas antenas para criar uma "cortina"de cobertura RFID [15].A antena pode ser vista na figura 3.4 a seguir.

Figura 3.4: Antena Impinj Threshold - Foto obtida no *datasheet* da antena [15]

A antena opera em duas faixas de frequência UHF: 902-928 MHz (FCC) e 865-868 MHz (ETSI), possui um ganho de campo distante de 5.0 dBi e impedância nominal de 50 Ω [15]. A antena cobre um volume elipsoidal de dimensões 3m x 4m x 3m, como pode ser visto na figura 3.5.

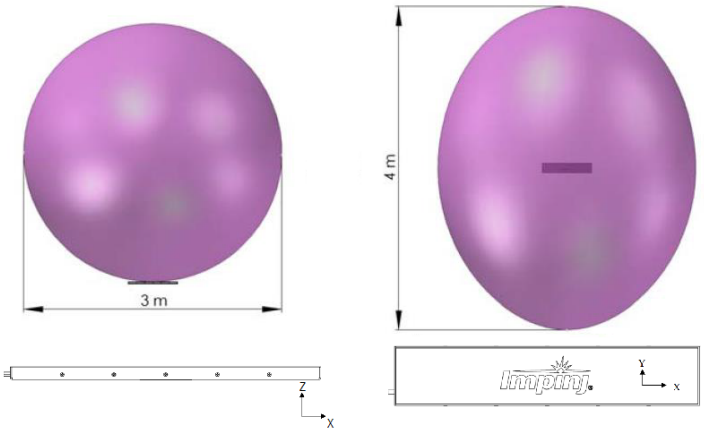


Figura 3.5: Gráfico da zona de cobertura das antenas Impinj Threshold e desenho mecânico da leitora em pontos de vista iguais aos dos desenhos de cobertura acima - Adaptado do *datasheet* da antena [15]

3.1.1.3 TAG utilizada no projeto

As TAGs utilizadas são impressas em adesivos pela empresa Nycti BR. Elas utilizam chip Alien Higgs-3 RFID IC. Como características principais estão:

- Banda de operação de 915-868 Mhz;
- Tamanho 96 X 23mm;
- Alcance de leitura de acordo com o fabricante de 6m (com testes empíricos feitos nas TAGs utilizadas de alcance de mais de 10m);
- 800-bit de memória, sendo 96 bits para armazenar o EPC - extensível até 480 bits - e 512 bits de uso de propósito geral; [42]
- ativado em baixíssimas potências de alimentação, ainda provendo excelente sinal de resposta de retroespalhamento. [42]

A figura 3.6 mostra as TAGs utilizadas para monitorar o trânsito das pessoas no interior dos ambientes pre-estabelecidos.

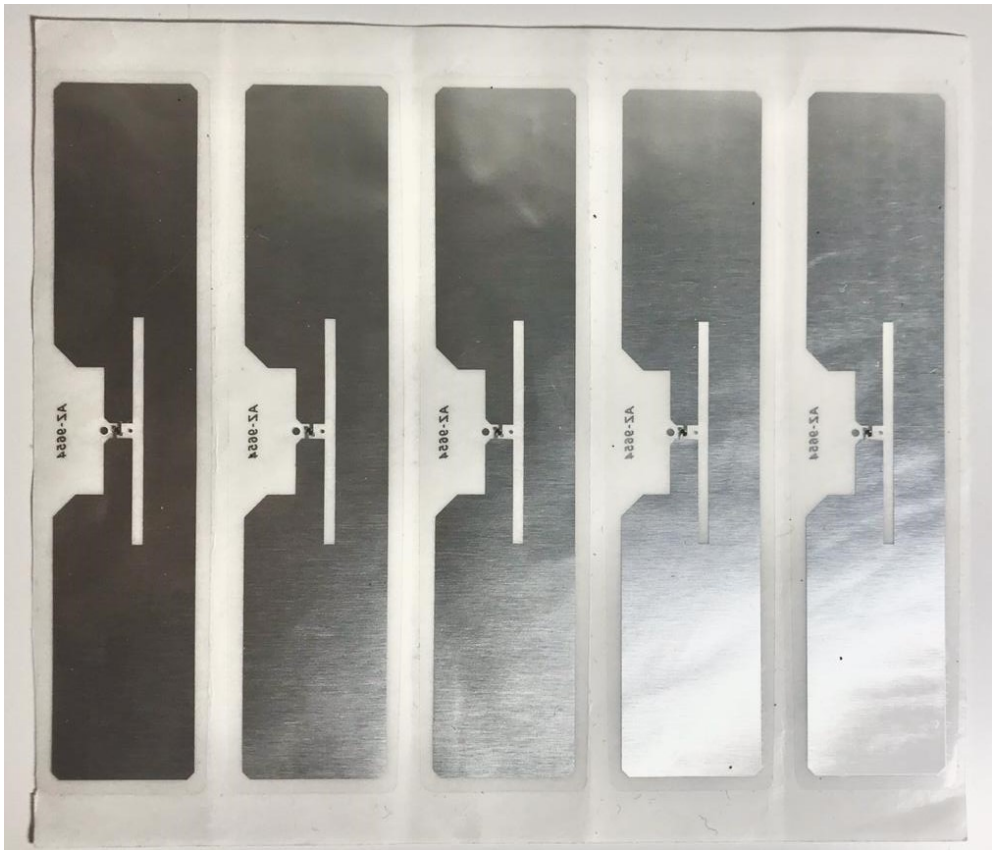


Figura 3.6: TAGs utilizadas para a execução deste trabalho

Estas TAGs mostradas na figura 3.6 foram distribuídas entre as pessoas designadas para testar o sistema criado.

3.1.1.4 Quantitativo

Este trabalho conta com leitoras e antenas RFID, além do espaço para trabalho e testes, gentilmente cedidos pelo Laboratório de Automação e Robótica (LARA) da UnB. No que se refere aos equipamentos, foram utilizados os seguintes dispositivos:

- 3 (três) Leitoras IMPINJ Speedway R420 (cedidas pelo LARA)
- 6 (seis) Antenas IMPINJ Treshold (cedidas pelo LARA)
- 1 (um) hub DLINK 6 portas Ethernet
- 20 (vinte) TAGs RFID 915 MHZ
- cabeamento e fontes de alimentação necessários

3.1.2 Software

Nesta parte do texto foram apresentadas as ferramentas computacionais utilizadas, suas características principais e como foram úteis na produção deste trabalho.

3.1.2.1 OctaneSDK

O pacote de desenvolvimento OctaneSDK para dispositivos RAIN RFID, da Impinj, foi utilizado para desenvolver a parte de software deste trabalho. O pacote possui versões para .NET e Java [43]. O pacote para .NET foi lançado há mais tempo e possui mais versões, e, por conseguinte, mais defeitos foram corrigidos, o que torna esta implementação mais robusta do que o pacote em Java. Por esse motivo, foi escolhido para o desenvolvimento desse trabalho.

3.1.2.2 Visual Studio

O pacote OctaneSDK .NET foi desenvolvido para funcionar com o Ambiente Integral de Desenvolvimento (*Integrated Development Environment* - IDE) Visual Studio, da Microsoft. A IDE foi utilizada para instalar o pacote de extensão do OctaneSDK. [43]

A linguagem de programação mais recomendada para desenvolvimento usando OctaneSDK com o Visual Studio é C#. O desenvolvimento em outras linguagens baseadas em .NET *framework* é possível, mas sem suporte da Impinj.

3.2 Abordagem utilizada

Três pontos foram essenciais para a execução deste trabalho: a instalação física das leitoras, antenas e demais equipamentos no local de teste; o *software* de detecção de pessoas e cruzamento de fronteiras para a contagem de pessoas em um determinado ambiente; e a elaboração de casos de teste para validação da abordagem. Estes pontos serão descritos nas seções a seguir.

3.2.1 Instalação física

As leitoras e as antenas utilizadas neste trabalho possuem um grande campo de cobertura relativo a outros modelos de equipamentos RFID passivo. Entretanto, a cobertura de sinal anunciada pelo fabricante é válida para ambientes abertos e sem a presença de objetos que possam interferir no sinal, como: mesas, cadeiras, equipamentos e pessoas.

A interferência de sinais de radiofrequência por metais e líquidos é um problema conhecido na indústria. Levando em consideração este problema, um cenário fictício de um ambiente de trabalho é considerado, baseado em escritórios e laboratórios reais. Supõe-se que todos os trabalhadores e visitantes, carreguem consigo crachás de identificação, nos quais estão embutidos transpônderes RFID já existentes, utilizados no sistema de controle de acesso do escritório ou laboratório.

As pessoas que transitam neste ambiente utilizam seus crachás em cordões dependurados no pescoço. Cada crachá permanece próximo ao corpo do seu usuário, porém não preso a ele. Os crachás ficam na altura do tórax.

3.2.2 O local

O Laboratório de Automação e Robótica (LARA) foi o local para simulação do ambiente de teste do sistema criado, especificamente, a metade sul do laboratório, onde se encontram os ambientes de estudo geral e a sala de reuniões. A planta baixa do laboratório pode ser observada na figura 3.7.

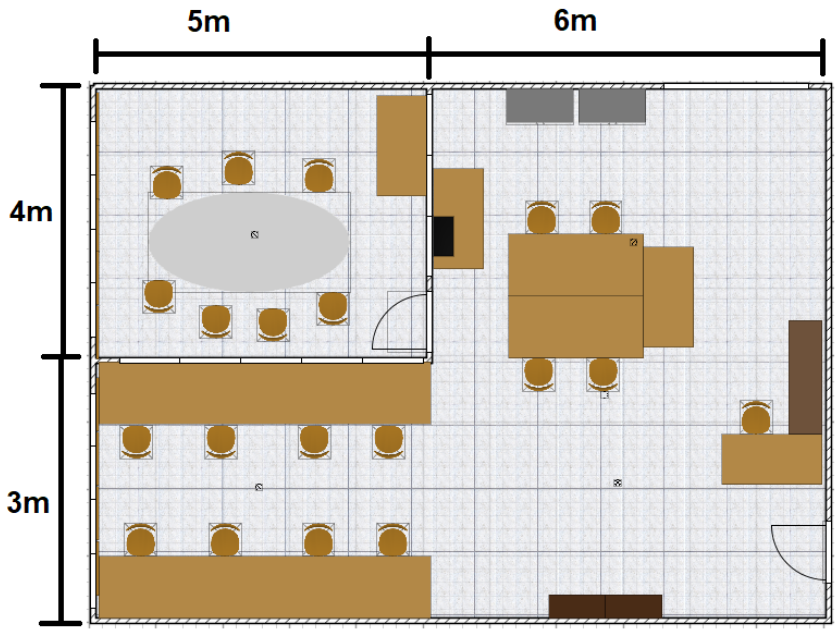


Figura 3.7: Imagem apresentando a planta baixa do LARA

Esta parte do laboratório foi subdividida em quatro ambientes: o Ambiente Externo (0), a Sala Principal (1), Sala de Reuniões (2) e o Corredor de Baias (3), como apresentado na marcação sobre a planta baixa, na figura 3.8.

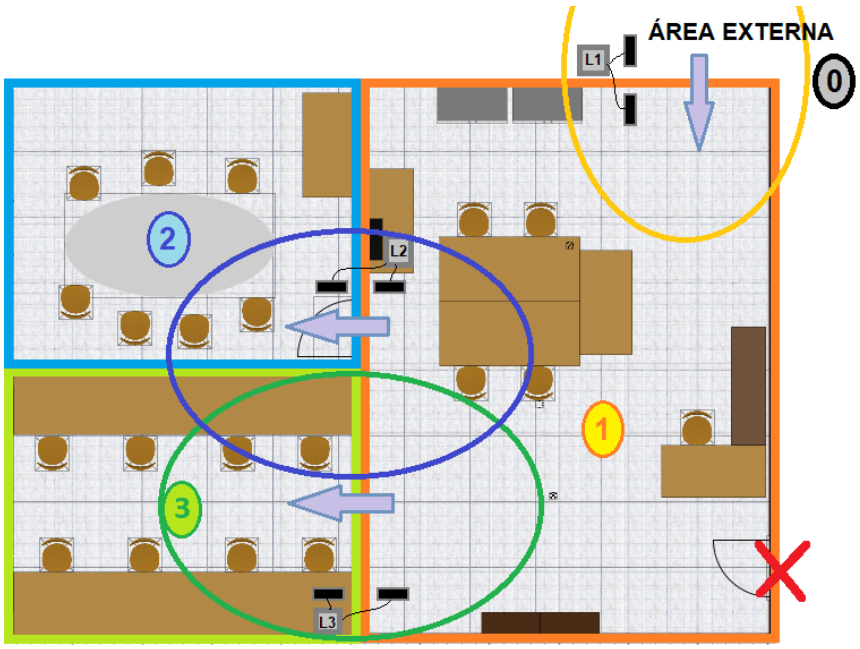


Figura 3.8: Imagem apresentando a planta baixa do LARA - local de instalação das leitoras

O ambiente do laboratório pode ser melhor visualizado nas figuras 3.9 e 3.10. As figuras mostram as projeções tridimensionais do laboratório, a partir de dois pontos de vista diferentes, para melhor compreensão do ambiente de estudo.

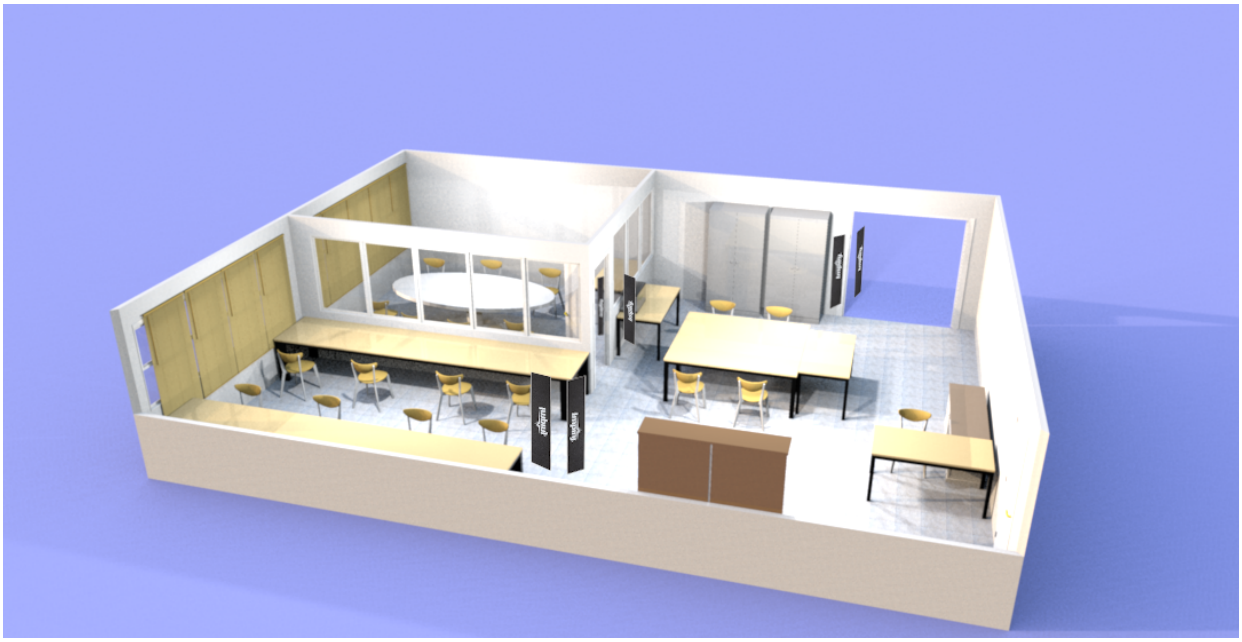


Figura 3.9: Imagem apresentando a renderização tridimensional do LARA e o local de instalação das leitoras

Figura 3.10: Imagem apresentando a renderização tridimensional do LARA a partir de um segundo ponto de vista - local de instalação das leitoras

As três leitoras foram designadas para tratar os dados das transições. A leitora 1 (L1) trata os dados da transição entre o Ambiente Externo (0) e a Sala Principal (1); a Leitora 2 (L2) é responsável por capturar os dados da transição entre a Sala Principal (1) e a Sala de Reuniões (2); a Leitora 3 registra os cartões que passam pela transição entre a Sala Principal (1) e o Corredor de Baias (3). Cada leitora é representada por um retângulo cinza na figura 3.8 e possui duas antenas para registrar as TAGs que passam na região.

As antenas foram dispostas em locais estratégicos para se obter as leituras mais precisas, nos locais de transição de interesse. Estes locais estão localizados próximos às portas ou passagens onde estão as fronteiras dos ambientes definidos. As antenas podem ser vistas como retângulos pretos de borda cinza na figura 3.8 e pelo desenho da leitora real nas figuras 3.9 e 3.9.

As leitoras da transição entre os ambientes Sala Principal (1) e Sala de Reuniões (2) e da transição entre os ambientes Sala Principal (1), Corredor de Baias (3) foram instaladas em posições opostas da sala para minimizar a ocorrência de leitura das TAGs pelas duas leitoras ao mesmo tempo. A marcação feita sobre a planta baixa, apresentada na figura 3.8, mostra em laranja, uma elipse estimando o alcance do campo de leituras da interrogadora L1, em azul, da leitora L2 e, em verde, a elipse de alcance da leitora L3.

3.2.3 Elaboração do *Software*

A tecnologia RAIN RFID, geralmente utilizada para aplicações de *IoT* (Internet das coisas - *Internet of Things* em inglês), exige a implementação de um *middleware* para conectar os dados crus das TAGs registrados pelas leitoras RFID às aplicações de alto nível e de computação em

nuvem, como pode ser observado na figura 2.4 do capítulo anterior, seção 2.2.1. Por esse motivo, o pacote OctaneSDK da Impinj foi escolhido para implementar a aplicação de localização de pessoas. O pacote possui classes prontas de comunicação e configuração das leitoras Impinj Speedway R420 usadas. Será considerado, portanto, que o programa criado não se trata de um *middleware*, mas de uma aplicação apenas, mesmo executando tarefas de comunicação com as leitoras.

O *software* elaborado simula internamente o ambiente do LARA como uma Rede de Petri, onde cada estado da rede representa um ambiente do LARA enquanto cada transição representa um par de antenas instaladas nas portas ou passagens. Os marcadores da Rede de Petri representam as TAGs das pessoas, transitando livremente pelos ambientes e transições. Um modelo que representa o funcionamento do programa pode ser visto na figura 3.11.

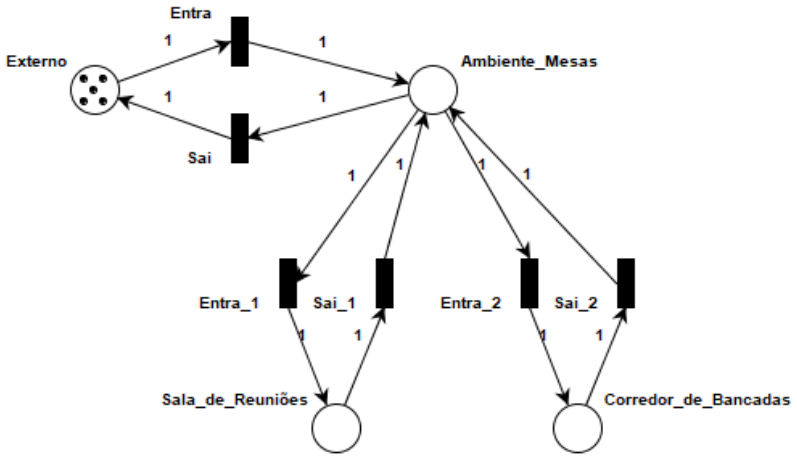


Figura 3.11: Rede de Petri representando todas as possibilidades circulação de pessoas pelo LARA como estados e transições

Simulando a Rede de Petri é possível observar a ideia da circulação das pessoas pelo espaço do laboratório, onde cada transição ativada leva um marcador de um estado a outro, o que corresponde a uma pessoa cruzando uma fronteira para ir de um ambiente a outro. A figura 3.12 mostra uma situação imaginária onde cinco pessoas transitam pelo laboratório. As transições marcadas em vermelho são as transições possíveis a partir do estado atual. A situação apresentada mostra duas pessoas na Sala Principal (1), duas pessoas na Sala de Reuniões (2) e uma pessoa no Corredor de Baias (3).

Os números próximos a cada seta entre um estado e uma transição nas figuras 3.11 e 3.12, neste caso todos iguais a 1, mostram o peso de cada transição. Dessa forma, a rede de Petri apresentada nas figuras possui igual probabilidade do acionamento de cada transição. A rede não necessariamente precisa ser implementada com uma entrada e uma saída para cada estado. É pos-

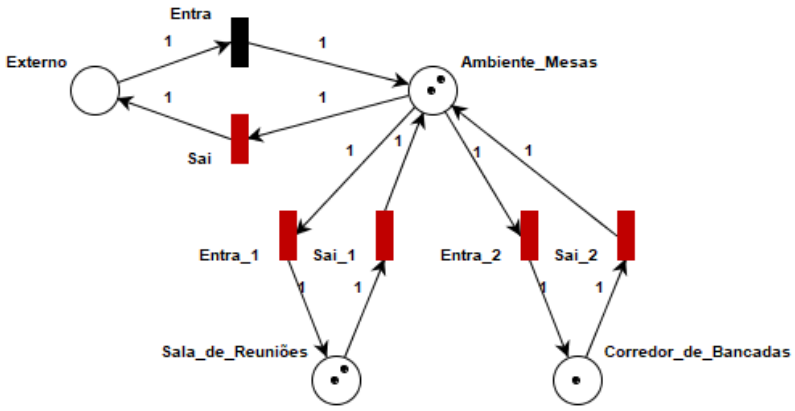


Figura 3.12: Rede de Petri representando a circulação de pessoas pelo LARA em um caso hipotético

sível encadear sequências em série, como o trajeto "*externo*" → "*entra*" → "*ambiente_mesas*" → "*entra_1*" → "*sala_de_reunioes*".

O programa criado em linguagem de programação C# é orientado a objeto e cria as seguintes classes:

- **Programa:** contém a função *main* e as funções básicas de configuração, comunicação e leituras;
- **Ambiente:** possui um nome, pode ter pessoas (*cardholders*) e possui uma carga térmica associada à quantidade de pessoas em seu interior;
- **Pessoa (*Cardholder*):** possui um nome, uma TAG (o número EPC associado à TAG) e uma carga térmica. Guarda a informação do ambiente em que se encontra e das curvas de histórico de leituras de valores RSSI e frequência Doppler;
- **Curva:** armazena os últimos n valores de RSSI ou frequência Doppler, assim como o horário das leituras em que se capturaram essas informações;
- **Antena:** possui um número e está associada a uma leitora;
- **Transição:** conecta dois ambientes, e monitora duas antenas;
- **Projeto:** implementa os critérios de decisão de transição e registra as mudanças de ambiente dos *cardholders*;
- **Gerenciador de arquivos (*FileHandler*):** Organiza e registra os dados coletados em arquivos CSV.

3.2.4 Configuração das Leitoras

A correta configuração das leitoras é provavelmente a parte mais importante deste trabalho. O estudo cuidadoso de todos os modos de operação possíveis, observando o manual do fabricante [14], *datasheet* das leitoras [39] e guias de programação e códigos de exemplo [43] resultou na escolha dos parâmetros apresentados nesta seção.

Primeiramente, conecta-se à cada leitora. Para isso, é necessário informar o caminho de rede para realizar a conexão. As leitoras foram configuradas com IP (*Internet Protocol*) fixo, em uma rede local fechada. O uso de uma rede local simplifica o comissionamento dos equipamentos e torna o sistema de localização de pessoas um pouco mais seguro, visto que RFID passivo é uma tecnologia muito suscetível a ataques à privacidade dos usuários e ataques destrutivos, como mostra Juels *et al.* [44]. Para acessar as leitoras, o nome padrão de domínio DNS (Sistema de Nomes de Domínio - em inglês *Domain Name System*) foi utilizado para facilitar a identificação das leitoras. O domínio DNS padrão das leitoras é composto pelo modelo, seguido dos três últimos pares de dígitos do número de série de cada uma, que estão impressos em etiquetas coladas a cada uma.

A figura 3.13 mostra a configuração dos caminhos de rede e a nomeação das leitoras. Através dos métodos *"new"* e *"Add"*, os objetos de cada leitora são criados e armazenados na classe *"readers"*.

```
// holds the paths to the readers
string hostname1 = "speedwayr-10-9f-3f.local";
string hostname2 = "speedwayr-10-9f-c8.local";
string hostname3 = "speedwayr-10-9f-bb.local";

// Create two reader instances and add them to the List of readers.
readers.Add(new ImpinjReader(hostname1, "Reader #1"));
readers.Add(new ImpinjReader(hostname2, "Reader #2"));
readers.Add(new ImpinjReader(hostname3, "Reader #3"));
```

Figura 3.13: Comandos de linha de código para estabelecer as leitoras

Em seguida, a primeira ação a ser executada é a criação do mapa da sala, com ambientes e transições, como discutido nas sessões 3.2.2 e 3.2.3. Os ambientes criados são "Area_Externa(0)", "Sala_Principal(1)", "Sala_Reuniao(2)"e "Corredor_Baias(3)". As transições 1, 2 e 3 são criadas e associadas às respectivas leitoras e antenas, conforme montado fisicamente no laboratório. A implementação pode ser vista na figura 3.14.

```
//Create map of rooms
Project.RegisterNewAmbient(0, new Ambient("Area_Externa(0)"));
Project.RegisterNewAmbient(1, new Ambient("Sala_Principal(1)"));
Project.RegisterNewAmbient(2, new Ambient("Sala_Reuniao(2)"));
Project.RegisterNewAmbient(3, new Ambient("Corredor_Baias(3)"));

//Create map of transitions
Transition transition1 = new Transition(Project.GetAmbientInstance(0), "Reader #1", 1, Project.GetAmbientInstance(1), "Reader #1", 2);
Transition transition2 = new Transition(Project.GetAmbientInstance(1), "Reader #2", 2, Project.GetAmbientInstance(2), "Reader #2", 1);
Transition transition3 = new Transition(Project.GetAmbientInstance(1), "Reader #3", 1, Project.GetAmbientInstance(3), "Reader #3", 2);
Project.RegisterNewTransition(Tuple.Create<string, ushort>("Reader #1", 1), transition1);
Project.RegisterNewTransition(Tuple.Create<string, ushort>("Reader #1", 2), transition1);
Project.RegisterNewTransition(Tuple.Create<string, ushort>("Reader #2", 1), transition2);
Project.RegisterNewTransition(Tuple.Create<string, ushort>("Reader #2", 2), transition2);
Project.RegisterNewTransition(Tuple.Create<string, ushort>("Reader #3", 1), transition3);
Project.RegisterNewTransition(Tuple.Create<string, ushort>("Reader #3", 2), transition3);
```

Figura 3.14: Comandos de linha de código para estabelecer os ambientes e as transições

A principal função da leitora RFID é reportar as informações coletadas do campo. A figura 3.15 mostra as configurações de *report* feitas. As informações coletadas em cada leitura incluem: qual antena coletou a informação, o horário em que a TAG foi vista a primeira e a última vez (em cada *report*), a quantidade de vezes que a TAG foi vista (em cada *report*), a frequência Doppler e o pico de potência RSSI da leitura.

O modo de *report* escolhido foi o modo individual, ou seja, a leitora envia um relatório a cada TAG lida. Isso inutiliza algumas funções nativas das leitoras, como os horários em que a TAG foi vista a primeira e a última vez e a quantidade de vezes que a TAG foi vista, mas possibilita a operação em tempo real (em um sentido mais amplo, não estritamente rígido), que é essencial para a aplicação criada neste trabalho.

```
settings.Report.IncludeAntennaPortNumber = true;
settings.Report.IncludeFirstSeenTime = true;
settings.Report.IncludeLastSeenTime = true;
settings.Report.IncludeSeenCount = true;
settings.Report.IncludeDopplerFrequency = true;
settings.Report.IncludePeakRssi = true;
// Send a tag report for every tag read
settings.Report.Mode = ReportMode.Individual;
```

Figura 3.15: Comandos de linha de código para configurar o modo de *report* das leitoras

Após a configuração das leitoras, é necessário configurar as antenas (na realidade são as leitoras que são ajustadas nesta etapa, visto que as antenas são elementos passivos e fixos, mas os parâmetros alterados nesta parte são referentes ao funcionamento das antenas). Primeiramente são apagadas as configurações padrão e duas antenas são ativadas manualmente em cada leitora com potência máxima e sensitividade máxima. Existe a possibilidade de programar as leitoras para identificarem novas antenas automaticamente, todavia, para esta aplicação, onde a quantidade de antenas é fixa, não é necessário ativar essa função. O procedimento pode ser visto na figura 3.16.

```
//Settings de antena
settings.Antennas.DisableAll();
settings.Antennas.GetAntenna(1).IsEnabled = true;
settings.Antennas.GetAntenna(2).IsEnabled = true;
// Set all the antennas to the max transmit power and receive sensitivity
settings.Antennas.TxPowerMax = true;
settings.Antennas.RxSensitivityMax = true;
```

Figura 3.16: Comandos de linha de código para configurar o modo de operação das antenas

Finalmente, uma última função é ativada: o modo de leitura dos equipamentos é alterado para *"DenseReaderM8"*. Este modo ativa parâmetros internos da leitora para prover a maior quantidade de leituras com a maior confiabilidade possível. Este modo é modo de funcionamento mais lento, que provê leituras escassas, mas confiáveis. Ao ativar esse modo, boa parte das leituras devido a reflexão e falsas leituras são filtradas em *hardware*. O comando pode ser visto na figura 3.17.

```
settings.ReaderMode = ReaderMode.DenseReaderM8;
```

Figura 3.17: Comando de linha de código para configurar o modo de operação das leitoras

Após as configurações, as leitoras são iniciadas, uma *tread* de leitura e uma de processamento de dados são abertas, e as configurações estão finalizadas.

3.3 Taxa de amostragem

A partir da equação 2.7, calcula-se:

$$T_A = \frac{1}{900.000.000} * 8 * 800 * 1 = 0,000007 = 7\mu s \tag{3.1}$$

Na equação 3.1 considerou-se $M = 8$, pois o modo de leitura utilizado neste trabalho foi *"DenseReaderM8"*. A quantidade de Bits presentes na TAG utilizazda é 800. A interferência de múltiplas TAGs foi definida como 1, para que se calcule o período de amostragem mínimo. Isso resultou em $T_A = 7\mu s$.

Apesar dos cálculos, empiricamente, que somando os atrasos do *software* interno da Leitora, do *software* desenvolvido neste programa, das comunicações via *Ethernet* e dos casos onde múltiplas TAGs são lidas ao mesmo tempo (caso onde 20 TAGs chegam a atrasar a leitura em até 10 vezes o período normal para a leitura de cada TAG), o período de amostragem de TAGs em situações ideais fica entre $1ms$ e $30ms$.

Entretando, considerando ainda as obstruções de sinal do meio físico, a água no corpo do usuário do cartão, as múltiplas superfícies de reflexão do ambiente fechado do LARA e a interferência de sinal com as outras TAGs, antenas, demais fontes de sinal UHF do laboratório e com o próprio

sinal de cada TAG refletido nas paredes, chão e teto, o período de amostragem real pode chegar a superar 1s entre uma leitura e outra.

3.3.1 Transições

Com o intuito de definir com a maior precisão possível o momento de passagem de uma pessoa de um ambiente para outro foi decidido monitorar as portas e passagens com duas antenas. O objetivo é acumular dados de potência de *backscattering* das TAGs (RSSI) para possuir um valor comparativo de proximidade da TAG para cada antena. Além do sinal de potência RSSI, também é coletada a frequência de efeito Doppler, que indica um valor proporcional a velocidade vetorial em direção ortogonal à leitora, e pode indicar se a pessoa se aproxima ou de afasta desta.

Durante testes preliminares, o analisou-se a variação dos valores de RSSI e frequência Doppler com uma única TAG e uma única antena. Os resultados deste teste podem ser encontrados na seção 4.1. Com base nesses resultados duas conclusões foram tiradas: as leituras de frequência de efeito Doppler podem ser facilmente divididas entre aproximação e afastamento de uma determinada antena e as leituras de potência RSSI aumentam proporcionalmente com a proximidade da TAG à antena, e diminuem proporcionalmente ao afastamento.

A partir dessa ideia foi criado o conceito de transição, onde duas antenas registram a passagem de uma pessoa, e comparam entre si os dados RSSI e de frequência Doppler para estimar o sentido do caminhar da pessoa (entrando ou saindo de um ambiente). Cada antena fica de um lado da fronteira (porta ou passagem).

Definindo-se o maior valor de RSSI durante todas as leituras na passagem de uma pessoa por uma fronteira, define-se esse como o momento em que a pessoa está mais próxima daquela antena. Realiza-se o mesmo procedimento para a segunda antena, do outro lado da fronteira. O ponto médio define o momento de transição de um ambiente para o outro. Da mesma forma, o momento em que uma pessoa deixa de aproximar-se e passa a se afastar de uma antena, este é o momento em que a pessoa está mais próxima da antena. Utilizando o mesmo critério do RSSI, o ponto médio entre o cruzamento na frente das duas antenas marca o momento de transição.

3.3.1.1 Curvas

Foram criados dois *buffers* para armazenar as últimas leituras de cada TAG. Cada um armazena uma lista ordenada de tuplas $< x, y >$, onde x é o tempo da leitura, e y é a magnitude. A lista é ordenada com base na chave x, para garantir que os dados são salvos em ordem cronológica. O primeiro *buffer* armazena as leituras de RSSI. O segundo armazena as leituras de frequência Doppler.Os *buffers* foram chamados de "curvas", por guardarem as curvas históricas das últimas n leituras para cada valor.

O tamanho das curvas, inicialmente foi definido em 100 valores. Este se mostrou um valor demasiado grande, pois armazena dados de mais de uma passagem anterior, e atrapalha a análise dos dados. Definiu-se então que as curvas armazenariam 5 a 20 valores, padronizado em 12 leituras,

por ser a faixa do tamanho médio da quantidade de leituras feitas em uma passagem a passos curtos em velocidade constante na frente das antenas.

Após encher o *buffer*, cada próxima leitura será armazenada no final da lista, após a retirada do primeiro elemento, tornando o *buffer* uma fila.

3.3.1.2 Coleta de dados

Os dados coletados pela leitora utilizada neste projeto são, por vezes, ruidosos ou inconstantes. Valores de potência RSSI, quando traçados em curvas, costumam apresentar *outliers* frequentemente, com valores muito acima ou muito abaixo do esperado em algumas leituras. Já os valores de variação de frequência por efeito Doppler possuem bastante ruído na faixa próxima de zero Hertz.

Dadas as inconstâncias nos dados, para fazer um tratamento prévio à execução dos casos de teste (seção 3.3.2), um filtro de média móvel de três posições foi aplicado em todos os dados coletados. Dessa maneira, os *outliers* são rejeitados, sem afetar severamente a curva de dados, por se tratar de um filtro de grau 3.

Para os valores de frequência Doppler, uma outra tática foi implementada, além do filtro de médias móveis: um filtro rejeita-bandas foi implementado, rejeitando, assim, todos os valores lidos entre $-0,5 \leq f_D \leq 0,5$. Esse filtro rejeita-banda funciona como um gerador de histerese para o método de transição de ambientes por efeito Doppler (seção 3.3.2.5), impedindo que haja múltiplas leituras para apenas uma transição.

3.3.2 Estratégias e casos de teste

Durante a fase de planejamento deste trabalho, inicialmente foi cogitado o uso de apenas uma leitora por transição, utilizando apenas as informações fornecidas pelas leitoras de potência de retorno do sinal das TAGs (RSSI) e frequência de efeito Doppler para definição da posição de uma TAG em um ambiente ou outro. Essa ideia foi promissora em um primeiro momento, e chegou a mostrar bons resultados, como pode ser visto na seção 4.1. A identificação de um cruzamento de fronteira era feita corretamente na maioria das vezes, e as leituras geralmente produziam dados suficientes para a correta identificação da transição.

Entretanto, o problema com esta estratégia é que no momento em que adicionam-se ambientes vizinhos, com outras leitoras e outras antenas próximas, pessoas circulando e uma quantidade maior de TAGs, começam a surgir leituras erradas nas zonas de transição. Por exemplo: No ambiente do LARA, ao realizar o monitoramento com apenas uma antena na transição entre a Sala Principal (1) e a Sala de Reuniões (2), e uma antena na transição entre a Sala Principal (1) e o Corredor de Baias (3), no momento em que se passa por uma, leituras também são captadas pela outra.

Devido aos caminhos múltiplos que os sinais RF podem tomar em um ambiente fechado devido à reflexão (do chão, teto, paredes e objetos próximos, como mesas, cadeiras, hastes de metal,

equipamentos, etc.), a leitura de potência RSSI pode ser enganosa e mostrar a transição em um ambiente, quando na verdade se encontra em outro. Além disso, ao ficar próximo de uma divisória, como a parede da Sala de Reuniões (2), em alguns casos, sinais podem ser captados em antenas de fora da sala, enganando o programa e registrando uma troca de sala para outro ambiente. Por esse motivo, uma topologia similar à usada em sistemas de gerenciamento de estoque e sistemas controle de carga e descarga de caminhões [45] foi considerada como uma alternativa mais robusta do que apenas a detecção da presença de uma TAG.

Disponibilizando duas antenas para cada transição, é possível obter informações mais confiáveis de leitura, e ainda possui a vantagem de saber o sentido da travessia. Dessa forma, diminui-se o risco de registrar a entrada ou saída de uma pessoa em um ambiente apenas por registrar leituras de proximidade a uma antena. As antenas foram posicionadas de maneira similar à observada na figura 3.18. Esta foto, apesar de não ser o local final, permite a visualização da orientação e espaçamento das antenas para registrar a travessia de uma pessoa em sua frente. As leitoras e antenas, instaladas em seu local definitivo, podem ser visualizadas nas figuras 4.2, 4.3, 4.4, 4.5, 4.6 e 4.7.

Figura 3.18: Montagem das antenas em local provisório. É possível visualizar a orientação e distanciamento das antenas.

Foram elaboradas duas ideias principais, que foram estendidas para seis, ao final do estudo. As duas principais são discutidas nas sessões 3.3.2.2 e 3.3.2.5, e as demais (sessões 3.3.2.1, 3.3.2.6, 3.3.2.3 e 3.3.2.4) são derivadas das suas primeiras.

3.3.2.1 Comparando a magnitude do último valor de RSSI capturado

Iniciando pela ideia mais simples: comparam-se as últimas leituras em ambas as antenas. O princípio é utilizar a informação de potência do sinal de retorno das TAGs para a leitora, o sinal RSSI, como indicador de proximidade de uma antena. Quanto mais próximo uma TAG está de

uma antena, maior é o sinal RSSI registrado. Quanto mais distante uma TAG está da antena, menor é o sinal. Testes com as TAGs utilizadas no projeto mostraram que, quando estão rentes às antenas, encostando nelas, o sinal RSSI registrado chega a ser maior que $-30dB$. Ao afastar a TAG da antena, com a face da etiqueta apontando para ela, na orientação mais favorável à leitura pela antena (devido à polarização do sinal), o sinal mais fraco registrado foi ligeiramente menor que $-70dB$. O fabricante indica que as antenas são capazes de registrar até $-90dB$ de sinal de retorno RSSI, mas com as TAGs utilizadas e em um ambiente poluído para aplicações de RF uma leitura tão fraca não existiu.

Considerando que uma pessoa passe em frente às leitoras da foto 3.18 da direita para a esquerda, ou da esquerda para a direita, comparando-se o valor da magnitude da última leitura RSSI, em teoria, o valor lido pela antena mais próxima será sempre o mais forte. Como as duas antenas estão próximas, durante a maioria das leituras as duas antenas enxergarão a TAG a sua frente.

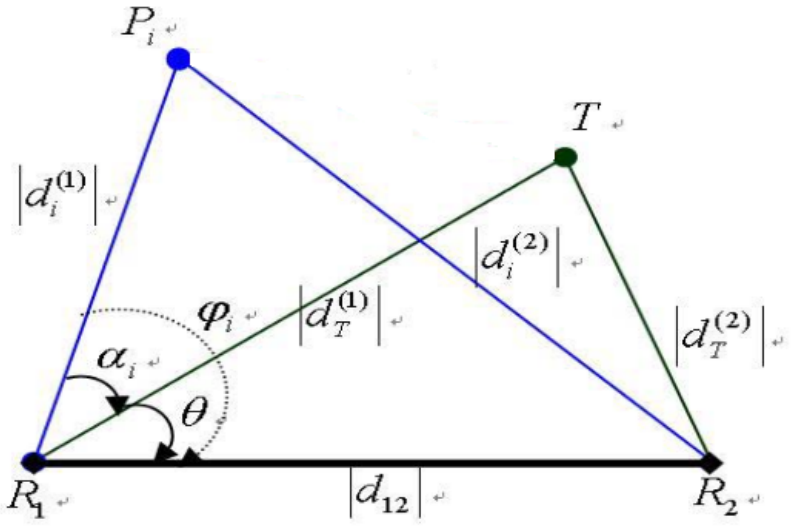


Figura 3.19: Esquemático das distâncias das TAGs às antenas (adaptado de Bekkali [16]

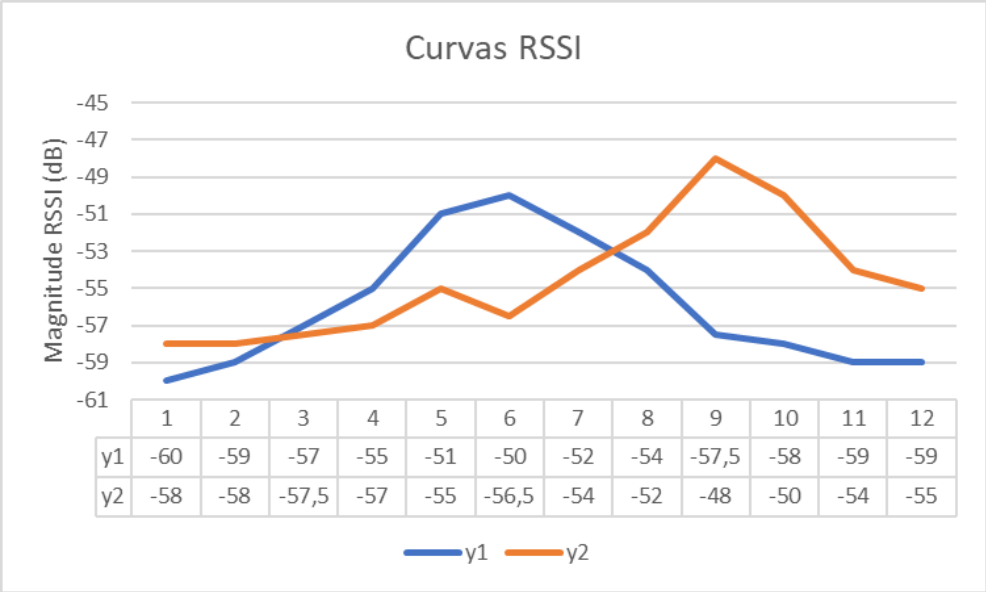
A figura 3.19 mostra duas posições P e T, que representam duas posições possíveis para uma TAG em frente às antenas R_1 e R_2. Como a distância de P para R_2 é maior que de P para R_1, o sinal RSSI será mais forte em R_1. Já T está mais próximo de R_2, e portanto o seu sinal RSSI será mais forte em R_2 do que em R_1.

3.3.2.2 Comparando o tempo dos últimos picos das curvas de RSSI

Aprimorando a ideia da seção 3.3.2.1, caso o valor das últimas leituras das antenas seja armazenado, mais informações podem ser obtidas do que apenas a proximidade instantânea das antenas.

Criando curvas que armazenam esses valores, como discutido na seção 3.3.1.1, é possível analisar toda a trajetória da pessoa em frente às antenas. Utilizando as curvas fictícias das figuras

3.20 e 3.21, é possível explicar a ideia da comparação de picos.



	1	2	3	4	5	6	7	8	9	10	11	12
y1	-60	-59	-57	-55	-51	-50	-52	-54	-57,5	-58	-59	-59
y2	-58	-58	-57,5	-57	-55	-56,5	-54	-52	-48	-50	-54	-55

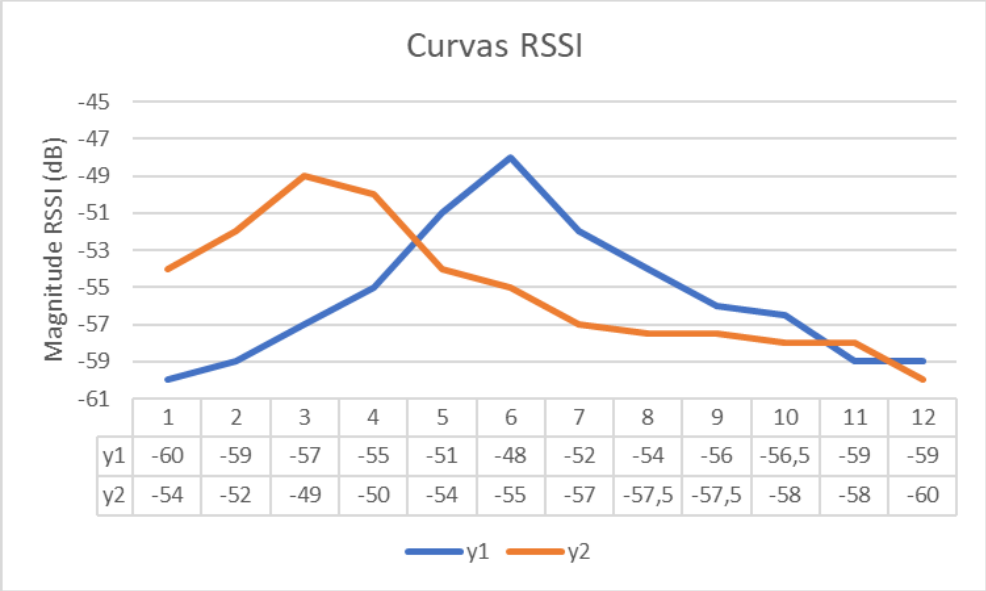
Figura 3.20: Exemplo de curvas RSSI com dados gerados artificialmente

Supondo que uma pessoa portando uma etiqueta RFID em seu crachá caminhe em frente às leitoras da figura 3.18. Definimos a antena da esquerda como a antena y_1, e a antena da direita como y_2. Ao caminhar da esquerda para a direita, a pessoa passa primeiramente em frente à antena y_1, e em seguida passa em frente à antena y_2. A cada período de tempo x - onde nesse caso teórico não atribuímos um valor de tempo definido para x a fim de facilitar o entendimento, apenas números inteiros de 1 a 12 - registra-se o valor de potência RSSI para a mesma TAG nas duas antenas nas curvas da figura 3.20, y_1 em azul e y_2 em laranja. Os tempos x de 1 a 12 mostram a progressão no tempo.

É possível perceber que entre os tempos $x = 5$ e $x = 7$ existe a maior probabilidade de a pessoa estar em frente à antena y_1, pois esse trecho apresenta os valores máximos de RSSI para essa antena. Entre os períodos $x = 8$ e $x = 10$ existe a maior probabilidade de a pessoa estar em frente à antena y_2, pois esse trecho apresenta os valores máximos de RSSI para essa antena. Já as curvas da figura 3.21 mostram o oposto: entre os períodos $x = 2$ e $x = 4$ a pessoa provavelmente passa em frente à antena y_2 e de $x = 5$ a $x = 7$ em frente à antena y_1.

A partir dessa informação, conclui-se que na situação da figura 3.20 a pessoa caminha da direita para a esquerda, em frente às leitoras. Já na figura 3.21, a pessoa caminha da esquerda para a direita. O intervalo entre o pico das duas curvas é o trecho entre as duas antenas. Define-se o ponto médio entre os dois picos como o ponto de cruzamento.

Um caso mais complicado onde a pessoa volta para o caminho de onde veio é mostrado na figura 3.22, também gerada com dados artificiais. Neste caso, dois picos são criados na curva azul, da antena y_1. Para tratar este tipo de imprevisto, o método da comparação dos tempos de pico também é robusto. Comparando os picos mais recentes, apenas, o critério de decisão enxergará um pico de potência RSSI de y_2 em $x = 6$ e um pico de potência RSSI de y_1 em $x = 10$, definindo



	1	2	3	4	5	6	7	8	9	10	11	12
y1	-60	-59	-57	-55	-51	-48	-52	-54	-56	-56,5	-59	-59
y2	-54	-52	-49	-50	-54	-55	-57	-57,5	-57,5	-58	-58	-60

Figura 3.21: Exemplo de curvas RSSI com dados gerados artificialmente

o ambiente onde a pessoa está como o lado da antena y_1, ou seja, no lado correto.

3.3.2.3 Comparando a média das curvas de potência RSSI

Dado que se possui um *buffer* repleto dos últimos dados de potência RSSI, obtidos pelas curvas armazenadas, considera-se a ideia de que a média desses valores possa indicar o lado da antena mais próxima. Estima-se que as últimas coletas de valores de RSSI na antena mais próxima, maiores que os da antena adjacente, possam fornecer bons indicadores de qual ambiente a pessoa ocupa.

A figura 3.23 mostra um comparativo entre duas curvas e suas médias, em uma situação hipotética onde uma pessoa permanece mais próxima da antena y_1 do que da antena y_2 por tempo suficiente para que as leituras de dados de RSSI preencham toda a curva.

3.3.2.4 Combinando a mediana das curvas de potência RSSI

Da mesma forma, aproveitando os dados das curvas, estima-se que a mediana dos dados seja maior do lado onde os últimos registros de leitura de TAG sejam feitos.

A figura 3.23 mostra um comparativo entre duas curvas e suas medianas, assim como as médias. O valor das medianas geralmente é diferente, mas não muito distante do valor das médias.

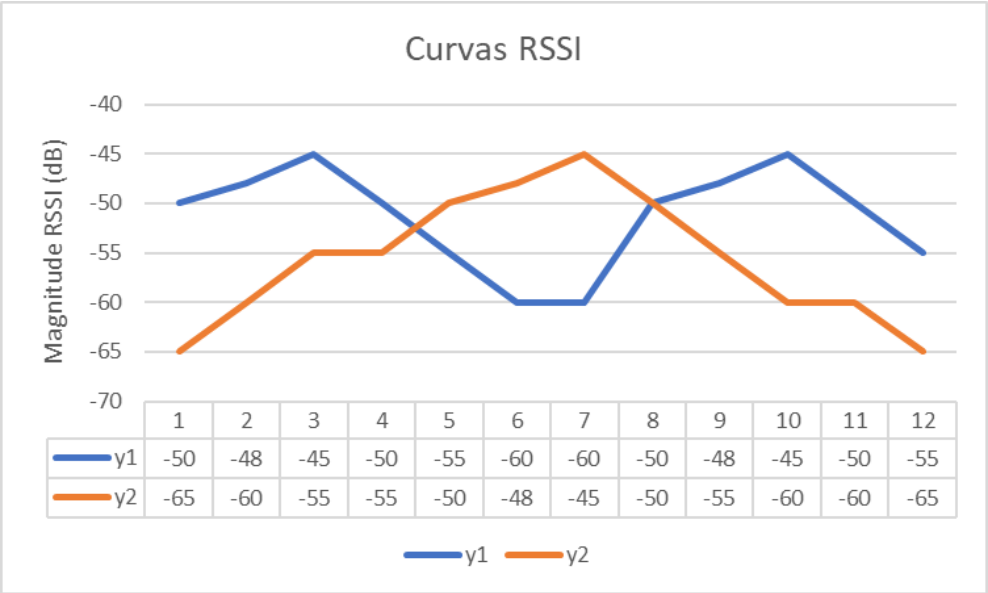


Figura 3.22: Exemplo de curvas RSSI com dados gerados artificialmente

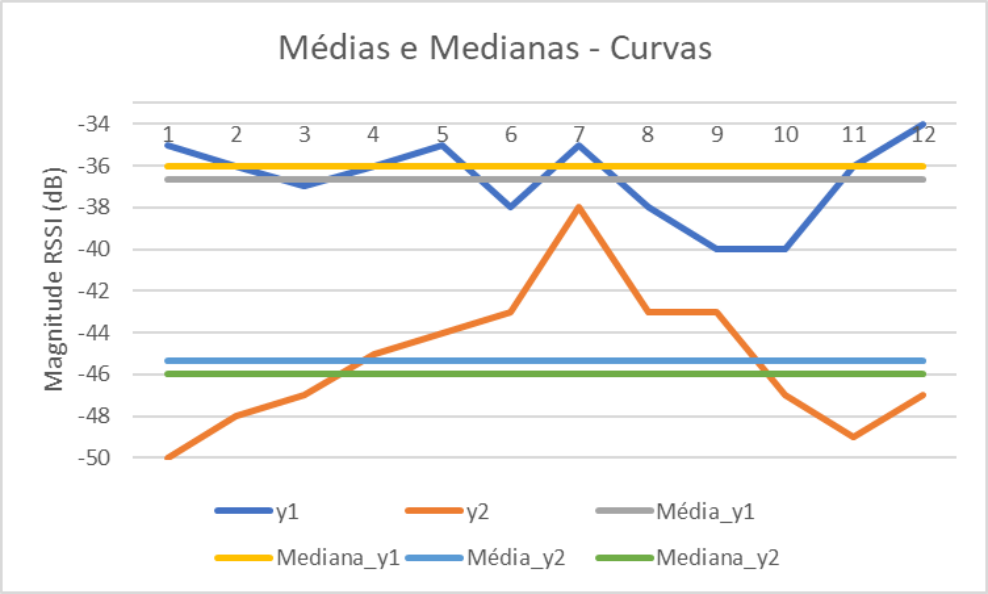


Figura 3.23: Exemplo de curvas RSSI com dados gerados artificialmente para cálculo de médias e medianas das curvas

3.3.2.5 Utilizando o Efeito Doppler como critério de travessia

O outro dado relevante para utilização como critério de rastreamento de pessoas é a frequência Doppler. Os dados fornecidos pelas leitoras indicam um número proporcional à componente de velocidade vetorial ortogonal à antena, como visto na seção 2.4. A figura 3.24 mostra dados gerados artificialmente de frequência Doppler simulando uma pessoa caminhando em velocidade constante em frente às antenas.

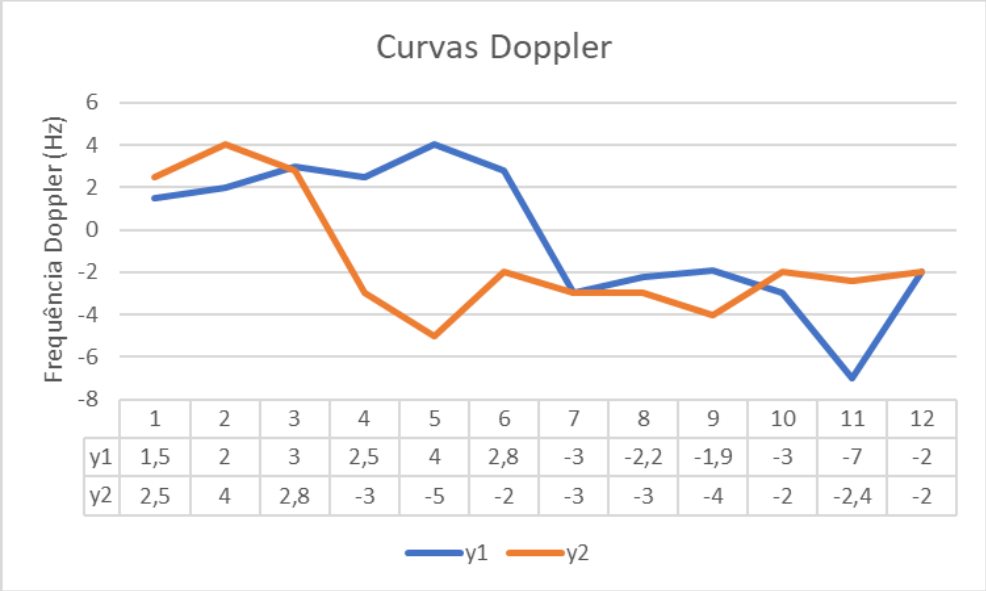


Figura 3.24: Exemplo de curvas Doppler com dados gerados artificialmente

Os dados fornecidos pela leitora, geralmente ficam na faixa entre $\pm 10Hz$. Valores maiores que zero indicam aproximação enquanto valores menores que zero indicam afastamento. Utilizando a equação 3.2 é possível normalizar os dados entre 1 e -1, criando um sinal binário (positivo ou negativo). A normalização dá origem à figura 3.25.

$$y_i = \frac{y_i}{|y_i|} \qquad ; i = 0, 1, 2, 3... \tag{3.2}$$

A figura 3.25 pode ser facilmente analisada pois com as curvas normalizadas, busca-se apenas as transições de positivo para negativo. Estas indicam que a pessoa deixou de aproximar-se e passou a se afastar da antena.

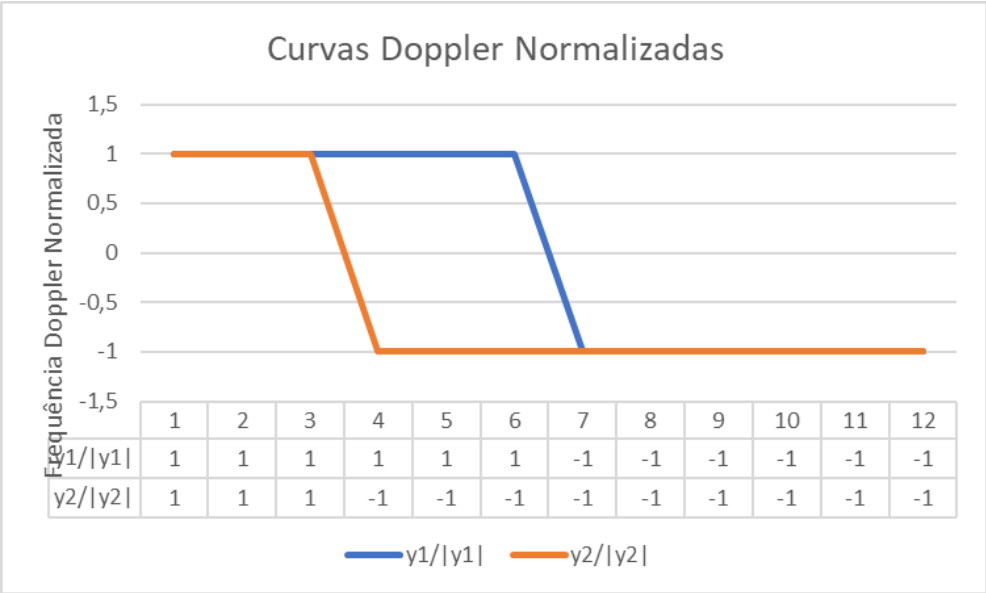


Figura 3.25: Exemplo de curvas Doppler normalizado com dados gerados artificialmente

Usando as transições das curvas normalizadas, procura-se por transições positivo→negativo consecutivas, pois a mais recente provavelmente indica o sentido em que a pessoa caminha. Por exemplo, a figura 3.25 possui a transição de y_2 em $x = 4$ e a de y_1 em $x = 7$, o que indicaria que a pessoa entrou no ambiente definido pela antena y_1.

3.3.2.6 Combinando a travessia por comparação de tempo de pico RSSI com o Efeito Doppler

O último caso estudado combina a comparação dos tempos de picos de potência de sinal RSSI e a comparação das transições positivo→negativo da frequência Doppler para confirmar com mais exatidão a existência de um cruzamento de fronteira. A figura 3.26 combina as figuras 3.20 e 3.25 em uma.

Na figura 3.26 observa-se que o pico $RSSI_1$ no instante $x = 6$ ocorre ao mesmo tempo que a última leitura positiva de frequência Doppler $\frac{FD_1}{|FD_1|}$, e o pico $RSSI_2$ no instante $x = 3$ ocorre ao mesmo tempo que a última leitura positiva de frequência Doppler $\frac{FD_2}{|FD_2|}$. Valores de pico de potência de sinal de resposta da TAG e transições de efeito Doppler acontecendo em momentos de tempo próximos indicam boa confiabilidade da sugestão de cruzamento de fronteira.

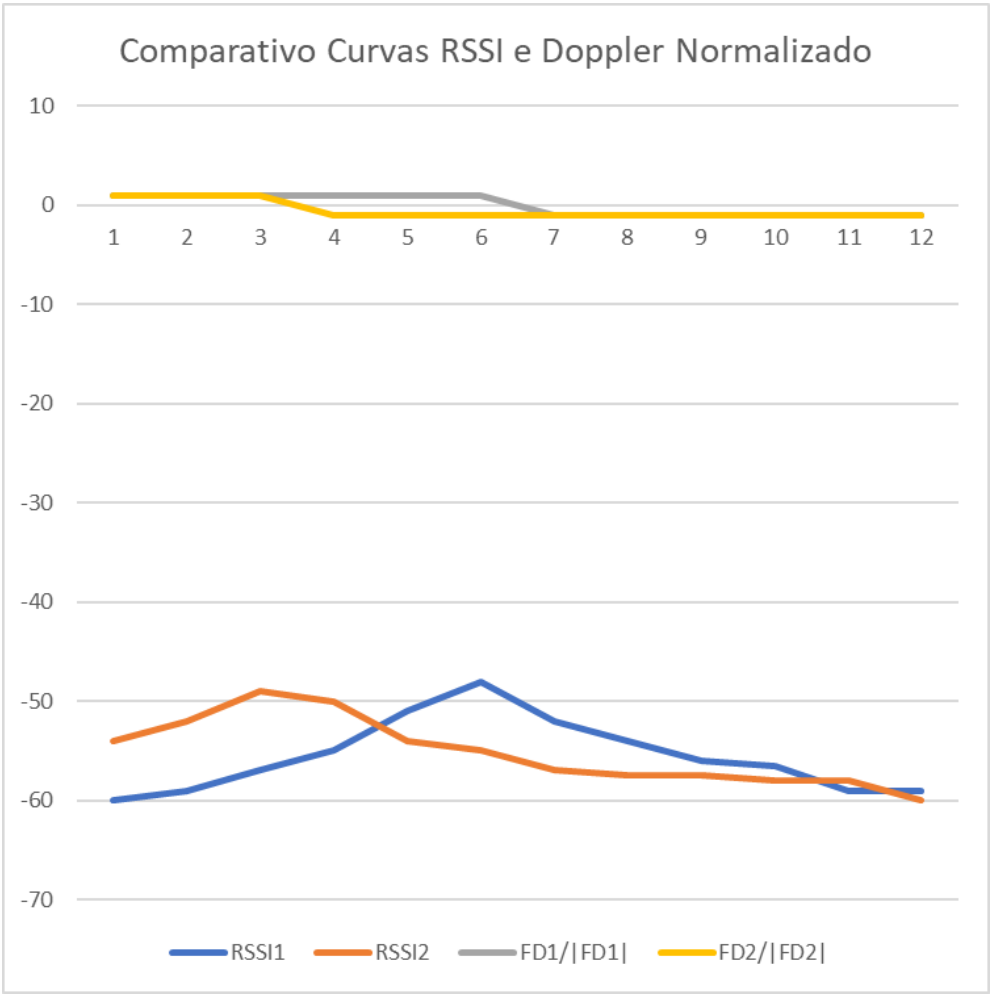


Figura 3.26: Comparativo das curvas RSSI e Doppler Normalizado com dados gerados artificialmente

Capítulo 4

Resultados

4.1 Teste preliminar

Durante testes preliminares, analisou-se a variação dos valores de RSSI e frequência Doppler com uma única TAG e uma única antena. A princípio as leituras pareciam muito sujeitas a ruído e pequenas alterações, então foram aplicados dois filtros: um filtro passa altas com corte em -60dB para o sinal de RSSI e um filtro rejeita banda para frequência Doppler para ignorar leituras menores que -1,50Hz e maiores que 1,50Hz. O teste preliminar resultou na figura 4.1.

O teste se mostrou bastante promissor, pois é possível observar na figura 4.1 que realizando as configurações corretas e aplicando filtros para rejeitar *outliers* de leitura é possível extrair informações úteis das leituras. Neste teste o valor da frequência Doppler foi observado para definir se a TAG se aproxima ou se afasta da antena.

Ainda é possível perceber que as leituras de frequência Doppler e RSSI, seguem tendências bem definidas. As leituras de frequência Doppler são positivas ao aproximar a TAG da antena, e negativas ao afastar a TAG da antena. A transição entre leituras positivas e negativas ocorre no exato momento de passagem na frente da leitora. As leituras de potência RSSI tentem a ser maiores no momento em que a pessoa está mais próxima da antena, e menores a medida em que se afasta da antena.

Apesar de seguirem uma tendência bem definida, possuem variações abruptas, algumas vezes no sentido oposto ao esperado. Por exemplo, observa-se uma queda abrupta de -51,00 dB e -52,50 dB para -57,00 dB nas duas transições entre o momento em que a pessoa se aproxima e o momento em que a pessoa se afasta da antena. Quanto ao efeito Doppler, este é proporcional à velocidade da pessoa, entretanto, o caminhar de uma pessoa não segue uma velocidade constante a todo momento, e portanto nota-se uma flutuação constante no valor lido.

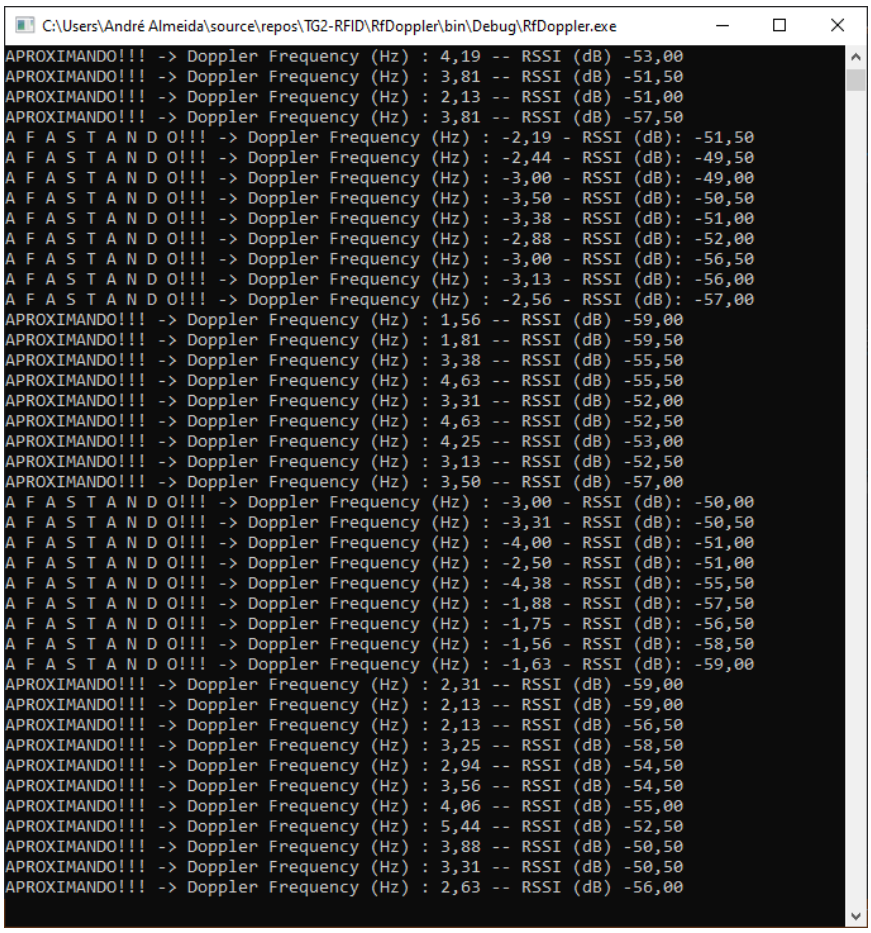


Figura 4.1: Captura de tela mostrando os dados de frequência Doppler e potência RSSI capturados no teste preliminar

4.2 Teste do programa implementado

O *software* criado para este trabalho segue a metodologia e as estratégias mencionadas na seção 3.3.2. O programa foi desenvolvido de tal forma que, ao mesmo tempo, é possível realizar todos os seis casos de teste. Para isso, a classe *cardholder* possui sete variáveis para armazenar ambientes: uma para o ambiente real - indicado manualmente no terminal do programa quando executado em modo *debug*, para uma pessoa específica apenas - e seis para as suposições de ambiente atual de cada caso de teste.

Um teste foi executado e, durante a execução, o programa armazenou, em tempo real, os dados relevantes de cada leitura de TAG (nome do portador da TAG, número EPC, nome da leitora, número da antena que capturou a leitura, data, horário da captura das informações, ambiente atual, curva de valores de potência RSSI, curva de frequência Doppler e as seis hipóteses de ambiente atual geradas pelos casos de teste). Estes dados podem ser vistos no anexo III.

Durante a execução do teste, uma pessoa transitou pelo ambiente do LARA, enquanto outra registrava manualmente os valores de ambiente real (anexo III).

As figuras 4.2, 4.3, 4.4, 4.5, 4.6 e 4.7 mostram imagens dos vídeos gravados, durante os cruza-

mentos de fronteira entre ambientes, em frente às leitoras e às antenas.

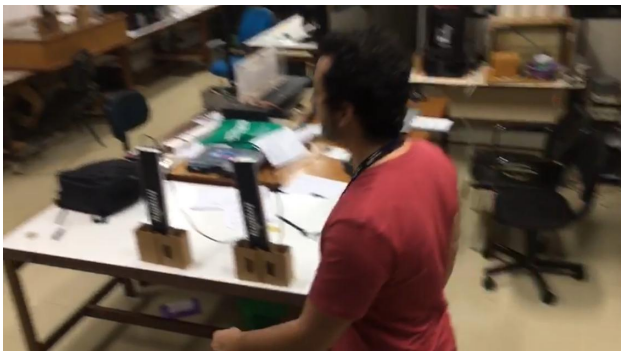


Figura 4.2: Teste do programa - Leitura do cruzamento da fronteira de Área Externa (0) para Sala Principal (1)

Figura 4.3: Teste do programa - Leitura do cruzamento da fronteira de Sala Principal (1) para Área Externa (0)

Figura 4.4: Teste do programa - Leitura do cruzamento da fronteira de Sala Principal (1) para Sala de Reuniões (2)

Figura 4.5: Teste do programa - Leitura do cruzamento da fronteira de Sala de Reuniões (2) para Sala Principal (1)

Figura 4.6: Teste do programa - Leitura do cruzamento da fronteira de Sala Principal (1) para Corredor Baias(3)

Figura 4.7: Teste do programa - Leitura do cruzamento da fronteira de Corredor Baias(3) para Sala Principal (1)

4.3 Eficiência dos métodos

Cada método de decisão de localização a seguir foi analisado manualmente de acordo com os dados gerados no anexo III. O critério para contabilizar um erro foi um método apresentar a sala incorreta mais de um minuto após a transição de ambientes ser contabilizada, ou até a próxima mudança de ambientes, o que ocorresse primeiro.

4.3.1 Comparação do último valor de RSSI

A eficiência do critério de último valor de RSSI baseada no caso de teste do anexo III pode ser vista na tabela 4.1

Tabela 4.1: Eficiência do critério do último valor com base no teste do anexo III

Eficiência do critério: Último valor de RSSI	
Erros confirmados	0/10
Confiabilidade	100%

Foram contabilizadas 10 transições e em todas elas, após no máximo quatro leituras, o presente método acertou o ambiente em que a pessoa se encontrava. O programa utiliza a localização de cada *cardholder* para contabilizar todas as pessoas em cada ambiente.

4.3.2 Comparação dos tempos dos últimos picos de RSSI

A eficiência do critério dos tempos de pico de RSSI baseada no caso de teste do anexo III pode ser vista na tabela 4.2

Tabela 4.2: Eficiência do critério dos tempos de picos de RSSI com base no teste do anexo III

Eficiência do critério: Tempos de pico de RSSI	
Erros confirmados	0/10
Confiabilidade	100%

Os itens de erro e confiabilidade desta tabela são os mesmos do caso de teste de último valor, seção 4.3.1, já que foram executados no mesmo teste. Foram 10 transições de ambiente e nenhuma se encontrou equivocada após quatro leituras.

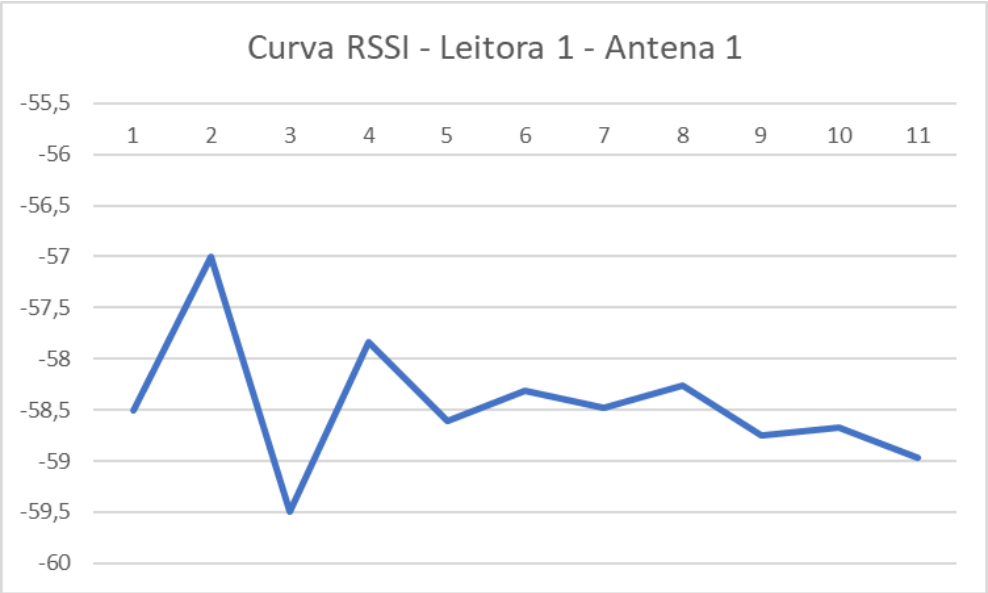


Figura 4.8: *Snapshot* de uma curva de valores RSSI no momento de uma transição - Curva da Leitora 1, Antena 1

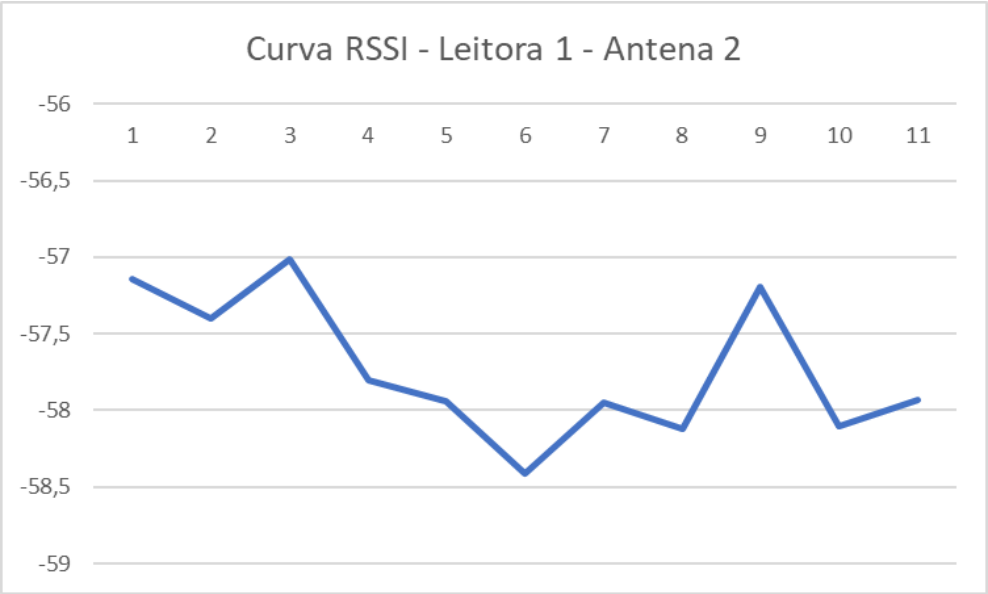


Figura 4.9: *Snapshot* de uma curva de valores RSSI no momento de uma transição - Curva da Leitora 1, Antena 2

As figuras 4.8 e 4.9 apresentam os gráficos de uma transição pelo critério dos picos RSSI. É possível observar o pico na curva da antena 2 (figura 4.9) no tempo 9 mais recente do que o último pico da antena 1 (figura 4.8), caracterizando uma passagem do lado da antena 2. Neste caso específico, realiza-se uma transição do Ambiente Externo (0) para a Sala Principal (1).

4.3.3 Comparação da média dos valores de RSSI

A eficiência do critério média dos valores de RSSI baseada no caso de teste do anexo III pode ser vista na tabela 4.3

Tabela 4.3: Eficiência do critério da média dos valores de RSSI com base no teste do anexo III

Eficiência do critério: Média dos valores de RSSI	
Erros confirmados	6/10
Confiabilidade	40%

O comparador de médias acertou 4 transições, entretanto, na maioria das transições, apresentou o valor errado. Analisando os dados gerados, acredita-se que o modo de funcionamento do programa, que mantém as curvas salvas indefinidamente, são facilmente enganadoras para este método. Ao passar por uma transição e, em um segundo momento, retornar, enquanto não houverem leituras suficientes para sobrescrever os dados das curvas, a média dos dados será muito próxima à ultima passagem. É necessário permanecer imóvel em frente a um par de antenas, obtendo leituras em ambas, para que a curva seja sobrescrita, e o método registre a transição.

4.3.4 Comparação da mediana dos sinais RSSI

A eficiência do critério mediana dos valores de RSSI baseada no caso de teste do anexo III pode ser vista na tabela 4.4

Tabela 4.4: Eficiência do critério da mediana dos valores de RSSI com base no teste do anexo III

Eficiência do critério: Mediana dos valores de RSSI	
Erros confirmados	7/10
Confiabilidade	30%

Os itens de erro e confiabilidade desta tabela são os similares aos resultados do caso de testes de média. Nenhum dos dois métodos foi considerado um método confiável ou aplicável.

4.3.5 Comparação da travessia por Efeito Doppler

A eficiência do critério de travessia por Efeito Doppler baseada no caso de teste do anexo III pode ser vista na tabela 4.5

Tabela 4.5: Eficiência do critério de travessia por Efeito Doppler com base no teste do anexo III

Eficiência do critério: Travessia por efeito Doppler	
Erros confirmados	3/10
Confiabilidade	70%

O caso de testes de contabilização de transição entre ambientes por efeito Doppler possui

resultados que o classificam como um estimador fazível, e que se deve levar em consideração. Entretanto, mostrou-se mais suscetível a erros causados por escarces de leituras. É válido um estudo mais aprofundado neste método, em um trabalho futuro, pois é um método promissor.

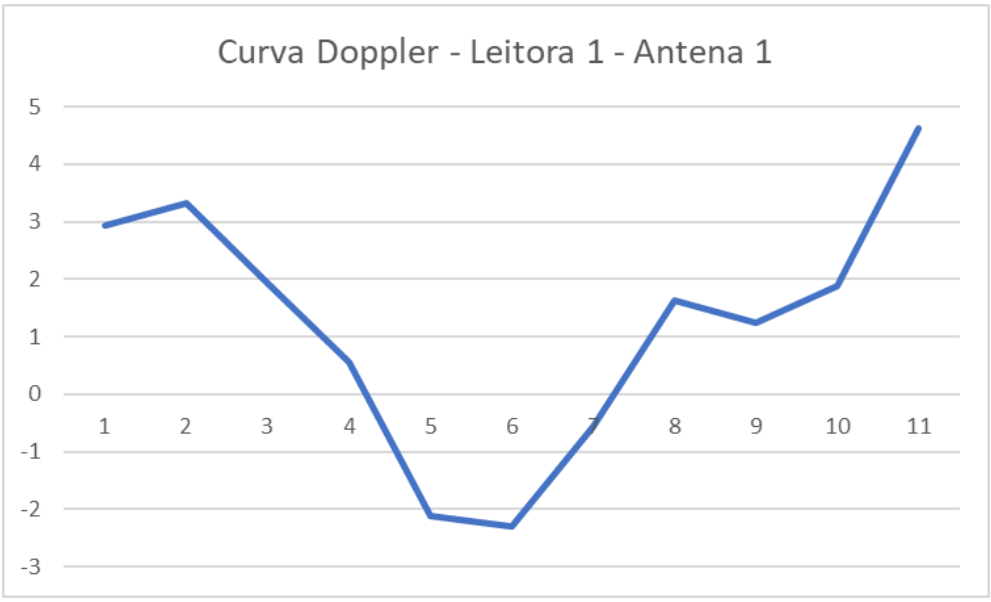


Figura 4.10: *Snapshot* de uma curva de frequências Doppler no momento de uma transição - Curva da Leitora 1, Antena 1

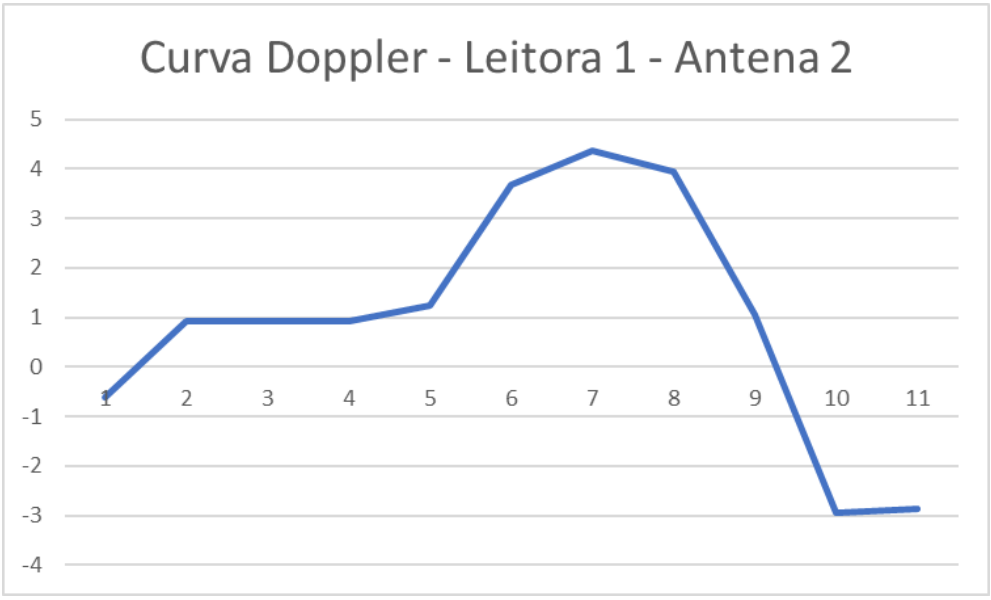


Figura 4.11: *Snapshot* de uma curva de frequências Doppler no momento de uma transição - Curva da Leitora 1, Antena 2

As figuras 4.10 e 4.11 apresentam duas curvas de frequências Doppler no momento da mesma transição observada nas figuras 4.8 e 4.9. É possível observar as transições positivo → negativo, no tempo 4 para a antena 1 e no tempo 9 para a antena 2, o que indica uma movimentação do

Ambiente Externo (0) para a Sala Principal (1), como no caso da seção 4.3.2. É interessante observar que estes são os mesmos tempos onde são observadas as curvas de pico do método que utiliza os valores de RSSI.

4.3.6 Combinação da técnica de picos de RSSI com Efeito Doppler

A eficiência do critério de travessia por tempos de pico RSSI e Efeito Doppler baseada no caso de teste do anexo III pode ser vista na tabela 4.6

Tabela 4.6: Eficiência do critério de combinação de comparação de picos de valores de RSSI e travessia por Efeito Doppler com base no teste do anexo III

Eficiência do critério: Travessia por efeito Doppler	
Erros confirmados	3/10
Confiabilidade	70%

Este método, da maneira como foi implementado, nivela os dois métodos de que é composto pelo pior desempenho. Isso se deve ao modo como foi implementado, aplicando-se a lógica "E"para combinar as sugestões de transição dos dois métodos. O teste com a combinação proporcional dos dois métodos, combinando os pesos para cada critério baseado nas suas respectivas eficiências.

Capítulo 5

Conclusões

Os resultados obtidos foram promissores para os critérios de decisão de comparação de picos de potência de sinal RSSI e de comparação do último valor de RSSI. Ambos obtiveram eficiência observada de 100%. Estes resultados, apesar de muito bons, podem ser validados de forma melhor com uma amostra de testes maior do que 10 transições. Dessa forma, novos testes com mais dados podem ser conduzidos.

Os resultados obtidos com o critério de transição por comparação de frequência Doppler foram abaixo do esperado, dado o resultado surpreendente do teste preliminar da sessão 4.1. Esperava-se uma eficiência maior que 80% de acerto, entretanto, este modelo obteve apenas 70% de acerto. Como há evidência de que este método pode funcionar com boa taxa de acerto no teste preliminar, provavelmente este método pode ser corrigido para obter maior eficiência.

O resultado obtido com a união do critério de comparação de picos e de efeito Doppler teve rendimento abaixo do esperado, com apenas 70% de eficiência, nivelando a eficiência dos dois métodos dos quais é composto pelo pior caso. Isso provavelmente se deve ao fato de que ele necessita que tanto o pico de RSSI quanto a transição de valores positivos para negativos do Efeito Doppler aconteçam ao mesmo tempo para registrar uma transição. Caso a leitora perca um dos eventos, este método não contabiliza a mudança de ambiente. Este método pode ser flexibilizado para aceitar ambos os critérios com lógica "OU"ao invés de lógica "E". Isso permitiria, talvez, uma taxa de acerto muito maior do que os outros métodos.

Os testes de mediana e média foram feitos sem muitas expectativas, e renderam resultados pouco expressivos - ambos obtiveram rendimento abaixo de 50%. Estes dois métodos são muito lentos para a velocidade média com que uma pessoa atravessa uma porta. Eles possuem potencial, entretanto, como métodos de correção para os outros, em casos onde todo o ambiente possui cobertura de sinal pelas antenas, pois sua natureza lenta acumula leituras repetidas durante longos períodos de tempo, e podem ser utilizados para detectar uma pessoa parada no interior de uma sala.

Este trabalho possui grande potencial de implementação em sistemas reais de ERP,EPC, especialmente em ambientes diferenciados, tais como a área cirúrgico-laboratorial. Esse tipo de ambiente exige um controle fino da temperatura, integrando inteligência artificial para controle

preditivo, para que os *setpoints* de condições ambientais sejam sempre atingidos em tempo mínimo e com a menor variação possível. Além disso, exigem um severo sistema se segurança e controle de acesso, onde o uso das mãos é indesejado por se tratar de ambientes com risco biológico envolvido.

Considerando os fins acadêmicos e de desenvolvimento tecnológico, o trabalho desenvolvido já possui grande valor, pois possui potencial para aprimoramento e possui diversas aplicações possíveis diferentes. Dado o sucesso dos resultados obtidos com os métodos de comparação do último valor RSSI e dos tempos de picos de potência do sinal de retorno, sistemas de ar condicionado específicos podem ser controlados utilizando dados gerados por aplicações como o programa desenvolvido. Alguns exemplos de aplicações diferentes do setor hospitalar são áreas de grande movimentação de pessoas (como estações de metrô - onde todos já possuem um bilhete com etiqueta RFID passiva embutida), ou locais onde o controle acurado de temperatura é o critério mais importante, como um *datacenter* ou laboratório de química.

5.1 Perspectivas Futuras

Este trabalho inovou nos métodos de identificação de mudança de ambiente em relação aos seus antecessores [9] [23]. A nova abordagem na aplicação e nos algoritmos abre espaço para a aplicação de novas técnicas. Em especial destacam-se a lógica difusa, ou lógica *fuzzy* [46], e a implementação de filtros preditivos, como o filtro de Kalman [47].

A lógica *fuzzy* pode ser usada para aprimoramento do critério de decisão sobre em qual lado da porta ou passagem o portador da TAG se encontra. Já os filtros preditivos, seja o filtro de Kalman ou outro, podem ser úteis na hora de coletar os dados para obter leituras mais precisas e acuradas, tornando o processo de decisão mais confiável.

É aconselhado também buscar outras abordagens, como o LANDMARK [16], que tem o intuito de criar um mapa do ambiente em que se deseja monitorar as pessoas, e por meio dos padrões de magnitude dos sinais estima o local exato dentro de uma sala onde a pessoa se encontra.

Para finalizar, o desenvolvimento de uma integração com sistemas ERP,EPC é altamente recomendado, para que os estimadores de localização e contabilização da quantidade de pessoas presentes possam ser integrados em sistemas supervisórios comerciais, sistemas de controle de ar condicionado e sistemas de controle de acesso.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] ENTERPRISE Security Applications. http://www.3dsecurity.ie/3d-security/Main/Enterprise_Security_Applications.html. [Online; Acessado em: 08 de dezembro de 2019].

[2] EPC-RFID INFO. *What is RFID?* 2019. https://www.epc-rfid.info/rfid. [Online; Acessado em: 04 de novembro de 2019].

[3] LANDT, J. The history of rfid. *IEEE potentials*, IEEE, v. 24, n. 4, p. 8–11, 2005.

[4] CHAWLA, V.; HA, D. S. An overview of passive rfid. *IEEE Communications Magazine*, IEEE, v. 45, n. 9, p. 11–17, 2007.

[5] CHEN, R. S. et al. Using rfid technology in food produce traceability. *WSEAS Transactions on information science and applications*, p. 1551–1560, 2008.

[6] WAVETREND. *Active RFID*. https://wavetrend.net/project_category/active-rfid/. [Online; Acessado em: 08 de dezembro de 2019].

[7] IDTECHEX. *Passive RFID grows by 1.12 billion tags in 2014 to 6.9 billion | IDTechEx Research Article*. https://www.idtechex.com/fr/research-article/passive-rfid-grows-by-1-12-billion-tags-in-2014-to-6-9-billion/7031. [Online; Acessado em: 08 de dezembro de 2019].

[8] RFID INSIDER. *Active RFID vs. Passive RFID: What's the Difference?* https://blog.atlasrfidstore.com/active-rfid-vs-passive-rfid. [Online; Acessado em: 08 de dezembro de 2019].

[9] OLIVEIRA, F. R. d.; ROCHA, F. S. d. P. *RFID PASSIVA NO RASTREAMENTO DE USUÁRIOS PARA A AUTOMAÇÃO PREDIAL*. 112 p. Monografia (Graduação) — Universidade de Brasília. Faculdade de Tecnologia, Distrito Federal, 2013.

[10] THORNTON, F.; SANGHERA, P. *How to cheat at deploying and securing RFID*. [S.l.]: Syngress, 2011.

[11] BUFFI, A. et al. Rssi measurements for rfid tag classification in smart storage systems. *IEEE Transactions on Instrumentation and Measurement*, IEEE, v. 67, n. 4, p. 894–904, 2018.

[12] TESCH, D. A.; BERZ, E. L.; HESSEL, F. P. Rfid indoor localization based on doppler effect. In: IEEE. *Sixteenth International Symposium on Quality Electronic Design*. [S.l.], 2015. p. 556–560.

[13] READER Modes Made Easy. https://support.impinj.com/hc/en-us/articles/360000046899-Reader-Modes-Made-Easy. [Online; Acessado em: 08 de dezembro de 2019].

[14] IMPINJ. *SpeedwayR Installation and Operations Guide*. [S.l.].

[15] IMPINJ. *THRESHOLD ANTENNA - READER ANTENNA DATASHEET*. [S.l.].

[16] BEKKALI, A.; SANSON, H.; MATSUMOTO, M. Rfid indoor positioning based on probabilistic rfid map and kalman filtering. In: IEEE. *Third IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2007)*. [S.l.], 2007. p. 21–21.

[17] AHMED, A. et al. Integration of value stream mapping with rfid, wsn and zigbee network. *Applied Mechanics and Materials*, Trans Tech Publications, v. 465, p. 769–773, 2014.

[18] SANGHERA, P. *RFID+ Study Guide and Practice Exams*. 1. ed. [S.l.]: Syngress, 2007. ISBN 978-1-59749-134-1.

[19] ANSI/ASHRAE. *Standart 55-2017 - Thermal Environmental Conditions for Human Occupancy*. Tullie Circle, NE, Atlanta, GA 30329-2305, 2017.

[20] ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS (ABNT). *NBR 15.575-1: Edificações Habitacionais – Desempenho*. Rio de Janeiro:ABNT, 2013.

[21] ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS (ABNT). *NBR 15.220-1: Desempenho térmico de edificações*. Rio de Janeiro:ABNT, 2003.

[22] GUTIERREZ, R. M. V. et al. Complexo eletrônico: identificação digital por radiofrequência. Banco Nacional de Desenvolvimento Econômico e Social, 2005.

[23] ALVES, R. A.; CHUPEL, R. C. M. *Utilização de RFID passiva com fusão sensorial para detecção de usuários em ambientes prediais*. 75 p. Monografia (Graduação) — Universidade de Brasília. Faculdade de Tecnologia, Distrito Federal, 2015.

[24] RAO, K. V. S. An overview of backscattered radio frequency identification system (rfid). In: IEEE. *1999 Asia Pacific Microwave Conference. APMC'99. Microwaves Enter the 21st Century. Conference Proceedings (Cat. No. 99TH8473)*. [S.l.], 1999. v. 3, p. 746–749.

[25] MICROSOFT AZURE. *O que é middleware?* https://azure.microsoft.com/pt-br/overview/what-is-middleware/. [Online; Acessado em: 08 de dezembro de 2019].

[26] RAIN RFID. *What is RAIN?* https://rainrfid.org/about-rain/what-is-rain/. [Online; Acessado em: 11 de novembro de 2019].

[27] GUBBI, J. et al. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, Elsevier, v. 29, n. 7, p. 1645–1660, 2013.

[28] FINKENZELLER, D. K. *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication, Third Edition*. 3. ed. [S.l.]: Wiley, 2010. ISBN 978-0-470-69506-7,0470695064.

[29] TELECO. *Redes Wi-Fi I: Antenas*. https://www.teleco.com.br/tutoriais/tutorialredeswifi1/pagina_3.asp. [Online; Acessado em: 08 de dezembro de 2019].

[30] FLETCHER, R.; MARTI, U. P.; REDEMSKE, R. Study of uhf rfid signal propagation through complex media. In: IEEE. *2005 IEEE Antennas and Propagation Society International Symposium*. [S.l.], 2005. v. 1, p. 747–750.

[31] FRANCêS, C. R. L. *Introdução às Redes de Petri*. https://www.dca.ufrn.br/~affonso/FTP/DCA409/redes_de_petri.pdf. [Online; Acessado em: 08 de dezembro de 2019].

[32] CARDOSO, J.; VALETTE, R. *Redes de Petri*. http://www2.ic.uff.br/~bruno/uploads/Lectures/Cardoso1997.pdf. [Online; Acessado em: 08 de dezembro de 2019].

[33] NIKITIN, P. V. et al. Phase based spatial identification of uhf rfid tags. In: IEEE. *2010 IEEE International Conference on RFID (IEEE RFID 2010)*. [S.l.], 2010. p. 102–109.

[34] BOUET, M.; SANTOS, A. L. D. Rfid tags: Positioning principles and localization techniques. In: IEEE. *2008 1st IFIP Wireless Days*. [S.l.], 2008. p. 1–5.

[35] JIN, G.-y.; LU, X.-y.; PARK, M.-S. An indoor localization mechanism using active rfid tag. In: IEEE. *IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC'06)*. [S.l.], 2006. v. 1, p. 4–pp.

[36] SANPECHUDA, T.; KOVAVISARUCH, L. A review of rfid localization: Applications and techniques. In: IEEE. *2008 5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*. [S.l.], 2008. v. 2, p. 769–772.

[37] IMPINJ xArray RFID Reader System Tracks In Real-Time. https://www.youtube.com/watch?v=eLgm9gdn0hg. [Online; Acessado em: 08 de dezembro de 2019].

[38] TESORIERO, R. et al. Using active and passive rfid technology to support indoor location-aware systems. *IEEE Transactions on Consumer Electronics*, IEEE, v. 54, n. 2, p. 578–583, 2008.

[39] IMPINJ. *Impinj Speedway Readers - Overview, Software Tools, Accessories and specifications*. [S.l.].

[40] GS1. *LLRP*. https://www.gs1br.org/codigos-e-padroes/epc-rfid/llrp. [Online; Acessado em: 09 de novembro de 2019].

[41] IEEE SA. *IEEE 802.3bt-2018 - IEEE Standard for Ethernet Amendment 2: Physical Layer and Management Parameters for Power over Ethernet over 4 pairs*. 2018. https://standards.ieee.org/standard/802_3bt-2018.html. [Online; Acessado em: 09 de novembro de 2019].

[42] ALIEN TECHNOLOGY. *Higgs™3 - EPC Class 1 Gen 2 RFID Tag IC*. [S.l.].

[43] IMPINJ. *Octane SDK Installation Instructions*. https://support.impinj.com/hc/en-us/articles/360010077479-Octane-SDK-Installation-Instructions. [Online; Acessado em: 10 de novembro de 2019].

[44] JUELS, A. et al. Rfid security and privacy: A research survey. *IEEE journal on selected areas in communications*, v. 24, n. 2, p. 381–394, 2006.

[45] PORTAL de carga RFID. `https://www.youtube.com/watch?v=IOZ64eWJQvQ`. [Online; Acessado em: 08 de dezembro de 2019].

[46] YEN, J.; LANGARI, R. *Fuzzy logic: intelligence, control, and information*. [S.l.]: Prentice hall Upper Saddle River, NJ, 1999.

[47] WELCH, G.; BISHOP, G. et al. An introduction to the kalman filter. Los Angeles, CA, 1995.

ANEXOS

I. DESCRIÇÃO DO CONTEÚDO DO CD

O CD contém uma cópia do Trabalho de Graduação 2 - MONITORAMENTO DE OCUPAÇÃO DE AMBIENTES USANDO RFID PASSIVO - em PDF e a pasta do projeto do programa desenvolvido em C#.

Os arquivos do CD estão organizados da seguinte forma, a partir da raiz:

- Monitoramento.PDF
- TG2-RFID (pasta)
  - Ambient.cs
  - Antenna.cs
  - Cardholder.cs
  - Curve.cs
  - FileHandler.cs
  - Program.cs
  - Project.cs
  - Transition.cs

II. PROGRAMAS UTILIZADOS

O programa desenvolvido em linguagem de programação C# foi apresentado neste Anexo. Os arquivos de cada classe do formato ".CS"foram organizados em ordem alfabética, seguindo o padrão:

- Ambient.cs
- Antenna.cs
- Cardholder.cs
- Curve.cs
- FileHandler.cs
- Program.cs (Main)
- Project.cs
- Transition.cs

```
1  using System;
2  using System.Collections;
3  using System.Collections.Generic;
4
5  namespace TG2_RFID
6  {
7      public class Ambient
8      {
9          /// <summary>
10         /// Holds the ambient name.
11         /// </summary>
12         protected string name;
13
14         /// <summary>
15         /// Holds the total thermal load.
16         /// A value that is proportional to the number of people in the room.
17         /// </summary>
18         protected int totalThermalLoad;
19
20         /// <summary>
21         /// Holds a map of people inside of room using their tag's EPC as key.
22         /// </summary>
23         protected Dictionary<string, Cardholder> cardholders;
24
25         /// <summary>
26         /// Holds the local antenna for this ambient.
27         /// </summary>
28         protected Antenna localAntenna;
29
30         /// <summary>
31         /// Constructor given the ambient name.
32         /// </summary>
33         public Ambient(string ambientName)
34         {
35             name = ambientName;
36             cardholders = new Dictionary<string, Cardholder>();
37         }
38
39         /// <summary>
40         /// Setter for the ambient antenna.
41         /// </summary>
42         public void SetAntenna(Antenna antenna)
43         {
44             localAntenna = antenna;
45         }
46
47         /// <summary>
48         /// Getter for the total thermal capacity in this room.
49         /// </summary>
50         public int GetTotalThermalLoad()
51         {
52             return totalThermalLoad;
53         }
54
55         /// <summary>
56         /// Getter for the total number of people inside of the ambient.
```

```
57          /// </summary>
58          public int GetNumberOfCarholdersInside()
59          {
60              return cardholders.Count;
61          }
62
63          /// <summary>
64          /// Adds a cardholder for this ambient.
65          /// </summary>
66          /// <param name="cardholder">Cardholder.</param>
67          public void AddCardholder(Cardholder cardholder)
68          {
69              try
70              {
71                  cardholders.Add(cardholder.GetTagEPC(), cardholder);
72              }
73              catch(Exception e)
74              {
75                  // Handle .NET errors.
76                  Console.WriteLine("Exception : {0}", e.Message);
77              }
78          }
79
80          /// <summary>
81          /// Removes a cardholder for this ambient.
82          /// If the cardholder is not in this ambient nothing happens.
83          /// </summary>
84          /// <param name="cardholder">Cardholder.</param>
85          public void RemoveCardholder(Cardholder cardholder)
86          {
87              if (cardholders.ContainsKey(cardholder.GetTagEPC()))
88              {
89                  cardholders.Remove(cardholder.GetTagEPC());
90              }
91          }
92
93          /// <summary>
94          /// Getter ambient name
95          /// </summary>
96          public string GetName()
97          {
98              return name;
99          }
100     }
101 }
102
```

```
1  using System;
2  using System.Collections;
3  using System.Collections.Generic;
4  using Impinj.OctaneSdk;
5
6  namespace TG2_RFID
7  {
8      public class Antenna
9      {
10         /// <summary>
11         /// Holds the source reader for this Antenna.
12         /// </summary>
13         protected ImpinjReader srcReader;
14
15         /// <summary>
16         /// Holds the port number of this Antenna.
17         /// </summary>
18         protected ushort srcAntennaPortNumber;
19
20         /// <summary>
21         /// Setter for the source reader for this antenna.
22         /// </summary>
23         /// <param name="newReader">New reader.</param>
24         public void SetReader(ImpinjReader newReader)
25         {
26             srcReader = newReader;
27         }
28
29         /// <summary>
30         /// Setter for the antenna port number.
31         /// </summary>
32         /// <param name="newAntennaPortNumber">New antenna port number.</param>
33         public void SetAntennaPortNumber(ushort newAntennaPortNumber)
34         {
35             srcAntennaPortNumber = newAntennaPortNumber;
36         }
37
38         /// <summary>
39         /// Getter for this antenna source reader.
40         /// </summary>
41         public ImpinjReader GetReader()
42         {
43             return srcReader;
44         }
45
46         /// <summary>
47         /// Getter for this antenna port number.
48         /// </summary>
49         public ushort GetAntennaPortNumber()
50         {
51             return srcAntennaPortNumber;
52         }
53
54         public Antenna()
55         {
56             srcReader = null;
```

```
57              srcAntennaPortNumber = 255;
58          }
59
60          /// <summary>
61          /// Constructor given the antenna reader and antenna port name.
62          /// </summary>
63          /// <param name="reader">Reader.</param>
64          /// <param name="antennaPortName">Antenna port name.</param>
65          public Antenna(ImpinjReader reader, ushort antennaPortName)
66          {
67              srcReader = reader;
68              srcAntennaPortNumber = antennaPortName;
69          }
70      }
71  }
72
```

```
 1  using System;
 2  using System.Collections;
 3  using System.Collections.Generic;
 4  using Impinj.OctaneSdk;
 5
 6
 7
 8  namespace TG2_RFID
 9  {
10      public class Cardholder
11      {
12
13          public static int RSSI_HIGHPASSFILTER = -60;
14          public static double DOPPLER_FILTER = 0.5;
15
16          /// <summary>
17          /// Holds the cardholder's name.
18          /// </summary>
19          protected string name;
20
21          /// <summary>
22          /// Holds the thermal capacity for this cardholder.
23          /// </summary>
24          protected int thermalCapacity;
25
26          /// <summary>
27          /// Holds the tag epc.
28          /// </summary>
29          protected string tagEPC;
30
31          /// <summary>
32          /// Holds whether the cardholder was initialized.
33          /// </summary>
34          protected bool wasInitialized;
35
36          /// <summary>
37          /// Holds the last seen tag as a Tag.
38          /// </summary>
39          protected Tag lastSeenTag;
40
41          /// <summary>
42          /// Holds the first seen time for this cardholder.
43          /// </summary>
44          protected DateTime firstSeenTime;
45
46          /// <summary>
47          /// Holds the last seen time for this cardholder.
48          /// </summary>
49          protected DateTime lastSeenTime;
50
51          /// <summary>
52          /// Holds the last seen rssi power value.
53          /// </summary>
54          protected double lastSeenRSSI;
55
56          /// <summary>
```

```
57          /// Holds the current ambient in which this cardholder is according to
               the default criteria.
58          /// </summary>
59          protected Ambient currentAmbient;
60
61          /// <summary>
62          /// Holds the current ambient in which this cardholder is according to
               the last RSSI peak time criteria.
63          /// </summary>
64          protected Ambient ambPeakTimes;
65
66          /// <summary>
67          /// Holds the current ambient in which this cardholder is according to
               the last RSSI peak time and magnitude criteria.
68          /// </summary>
69          protected Ambient ambPeakTimeAndMag;
70
71          /// <summary>
72          /// Holds the current ambient in which this cardholder is according to
               the last RSSI value criteria.
73          /// </summary>
74          protected Ambient ambLastRSSI;
75
76          /// <summary>
77          /// Holds the current ambient in which this cardholder is according to
               the last RSSI mean criteria.
78          /// </summary>
79          protected Ambient ambRSSIMean;
80
81          /// <summary>
82          /// Holds the current ambient in which this cardholder is according to
               the last RSSI median criteria.
83          /// </summary>
84          protected Ambient ambRSSIMedian;
85
86          /// <summary>
87          /// Holds the current ambient in which this cardholder is according to
               the Doppler Effect criteria.
88          /// </summary>
89          protected Ambient ambDopplerEffect;
90
91          /// <summary>
92          /// Holds the current ambient in which this cardholder is according to
               the Doppler Effect and RSSI criteria.
93          /// </summary>
94          protected Ambient ambDopplerAndRSSI;
95
96          /// <summary>
97          /// Holds a map of curves per antenna in which this cardholder was
              seen
98          /// holding the history of RSSI power per time.
99          /// </summary>
100         public Dictionary<Tuple<string, ushort>, Curve>
              curvesPowerReadingsDictionary;
101
102         /// <summary>
```

```
103         /// Holds a map of curves per antenna in which this cardholder was
              seen
104         /// holding the history of doppler frequency per time.
105         /// </summary>
106         public Dictionary<Tuple<string, ushort>, Curve>
              curvesDoplerFrequencyReadingsDictionary;
107
108         /// <summary>
109         /// Initializes a new instance of the <see
              cref="T:TG2_RFID.Cardholder"/> class.
110         /// </summary>
111         public Cardholder()
112         {
113             name = "DefaultName";
114             thermalCapacity = 1000;
115             wasInitialized = false;
116             curvesPowerReadingsDictionary = new Dictionary<Tuple<string,
                  ushort>, Curve>();
117             curvesDoplerFrequencyReadingsDictionary = new
                  Dictionary<Tuple<string, ushort>, Curve>();
118             ambDopplerAndRSSI = (Project.GetAmbientInstance(0));
119             ambDopplerEffect = (Project.GetAmbientInstance(0));
120             ambLastRSSI = (Project.GetAmbientInstance(0));
121             ambPeakTimeAndMag = (Project.GetAmbientInstance(0));
122             ambPeakTimes = (Project.GetAmbientInstance(0));
123             ambRSSIMean = (Project.GetAmbientInstance(0));
124             ambRSSIMedian = (Project.GetAmbientInstance(0));
125             currentAmbient = (Project.GetAmbientInstance(0));
126
127         }
128
129         /// <summary>
130         /// Initializes a new instance of the <see
              cref="T:TG2_RFID.Cardholder"/> class.
131         /// </summary>
132         /// <param name="personName">Person name.</param>
133         public Cardholder(string personName)
134         {
135             name = personName;
136             thermalCapacity = 1000;
137             wasInitialized = false;
138             curvesPowerReadingsDictionary = new Dictionary<Tuple<string,
                  ushort>, Curve>();
139             curvesDoplerFrequencyReadingsDictionary = new
                  Dictionary<Tuple<string, ushort>, Curve>();
140             ambDopplerAndRSSI = (Project.GetAmbientInstance(0));
141             ambDopplerEffect = (Project.GetAmbientInstance(0));
142             ambLastRSSI = (Project.GetAmbientInstance(0));
143             ambPeakTimeAndMag = (Project.GetAmbientInstance(0));
144             ambPeakTimes = (Project.GetAmbientInstance(0));
145             ambRSSIMean = (Project.GetAmbientInstance(0));
146             ambRSSIMedian = (Project.GetAmbientInstance(0));
147             currentAmbient = (Project.GetAmbientInstance(0));
148         }
149
150         /// <summary>
```

```
151         /// Initializes a new instance of the <see
              cref="T:TG2_RFID.Cardholder"/> class.
152         /// </summary>
153         /// <param name="personName">Person name.</param>
154         /// <param name="personThermalCapacity">Person thermal capacity.</
              param>
155         public Cardholder(string personName, int personThermalCapacity)
156         {
157             name = personName;
158             thermalCapacity = personThermalCapacity;
159             wasInitialized = false;
160             curvesPowerReadingsDictionary = new Dictionary<Tuple<string,
                  ushort>, Curve>();
161             curvesDoplerFrequencyReadingsDictionary = new
                  Dictionary<Tuple<string, ushort>, Curve>();
162             ambDopplerAndRSSI = (Project.GetAmbientInstance(0));
163             ambDopplerEffect = (Project.GetAmbientInstance(0));
164             ambLastRSSI = (Project.GetAmbientInstance(0));
165             ambPeakTimeAndMag = (Project.GetAmbientInstance(0));
166             ambPeakTimes = (Project.GetAmbientInstance(0));
167             ambRSSIMean = (Project.GetAmbientInstance(0));
168             ambRSSIMedian = (Project.GetAmbientInstance(0));
169             currentAmbient = (Project.GetAmbientInstance(0));
170         }
171
172         /// <summary>
173         /// Initializes a new instance of the <see
              cref="T:TG2_RFID.Cardholder"/> class.
174         /// </summary>
175         /// <param name="personName">Person name.</param>
176         /// <param name="personTagEPC">Person tag epc.</param>
177         public Cardholder(string personName, string personTagEPC)
178         {
179             name = personName;
180             thermalCapacity = 1000;
181             tagEPC = personTagEPC;
182             wasInitialized = false;
183             curvesPowerReadingsDictionary = new Dictionary<Tuple<string,
                  ushort>, Curve>();
184             curvesDoplerFrequencyReadingsDictionary = new
                  Dictionary<Tuple<string, ushort>, Curve>();
185         }
186
187         /// <summary>
188         /// Getter for the tag EPC.
189         /// </summary>
190         public string GetCardholderEPC()
191         {
192             return tagEPC;
193         }
194
195         /// <summary>
196         /// Setter for the tag EPC.
197         /// </summary>
198         /// <param name="newTagEPC">New tag epc.</param>
199         public void SetCardholderEPC(string newTagEPC)
```

```
200         {
201             tagEPC = newTagEPC;
202         }
203
204         /// <summary>
205         /// Getter for a RSSI power curve given an antenna.
206         /// </summary>
207         /// <returns>The power curve.</returns>
208         /// <param name="antenna">Antenna.</param>
209         public Curve GetPowerCurve(Tuple<string, ushort> antenna)
210         {
211             if (!curvesPowerReadingsDictionary.ContainsKey(antenna))
212             {
213                 curvesPowerReadingsDictionary.Add(antenna, new Curve());
214             }
215             curvesPowerReadingsDictionary.TryGetValue(antenna, out Curve
                retCurve);
216             return retCurve;
217         }
218
219         /// <summary>
220         /// Getter for the dopple effect curve given an antenna.
221         /// </summary>
222         /// <returns>The doppler effect curve.</returns>
223         /// <param name="antenna">Antenna.</param>
224         public Curve GetDopplerEffectCurve(Tuple<string, ushort> antenna)
225         {
226             if (!curvesDoplerFrequencyReadingsDictionary.ContainsKey(antenna))
227             {
228                 curvesDoplerFrequencyReadingsDictionary.Add(antenna, new Curve
                    ());
229             }
230             curvesDoplerFrequencyReadingsDictionary.TryGetValue(antenna, out
                Curve retCurve);
231             return retCurve;
232         }
233
234         /// <summary>
235         /// Getter for the tag EPC.
236         /// </summary>
237         /// <returns>The tag epc.</returns>
238         public string GetTagEPC()
239         {
240             return tagEPC;
241         }
242
243         /// <summary>
244         /// Getter for the person's name.
245         /// </summary>
246         public string GetName()
247         {
248             return name;
249         }
250
251         /// <summary>
252         /// TODO
```

```
253         /// TODO Reset buffer at 4am due to overflow problem.
254         /// </summary>
255         public void ReadingCardholderTag(Tag tag, string senderName)
256         {
257             lastSeenTag = tag;
258             lastSeenTime = DateTime.Now;
259             lastSeenRSSI = tag.PeakRssiInDbm;
260 
261 
262             if (!wasInitialized)
263             {
264                 firstSeenTime = lastSeenTime;
265                 wasInitialized = true;
266             }
267 
268             var diffTime = lastSeenTime - firstSeenTime;
269             double readingTime = (double)(diffTime.TotalMilliseconds);
270 
271             var tupleAntenna = Tuple.Create<string, ushort>(senderName,
                  tag.AntennaPortNumber);
272 
273             if (!curvesPowerReadingsDictionary.ContainsKey(tupleAntenna))
274             {
275                 curvesPowerReadingsDictionary.Add(tupleAntenna, new Curve());
276             }
277             var powerCurve = curvesPowerReadingsDictionary[tupleAntenna];
278             if (tag.PeakRssiInDbm > RSSI_HIGHPASSFILTER)
279             {
280                 powerCurve.AddPointWithAvgFilter(readingTime,
                      tag.PeakRssiInDbm);
281             }
282             // foreach (var antenna in curvesPowerReadingsDictionary.Keys)
283             //{
284             //    cnt++;
285             //    var powerCurve = curvesPowerReadingsDictionary[antenna];
286             //    if (tupleAntenna.Equals(antenna))
287             //    {
288             //        powerCurve.AddPointWithAvgFilter(readingTime,
                      tag.PeakRssiInDbm);
289             //    }
290             //    else
291             //    {
292             //        powerCurve.AddPointWithAvgFilter(readingTime, 0.0);
293             //    }
294             //}
295 
296 
297             if (!curvesDoplerFrequencyReadingsDictionary.ContainsKey
                  (tupleAntenna))
298             {
299                 curvesDoplerFrequencyReadingsDictionary.Add(tupleAntenna, new
                      Curve());
300             }
301             try
302             {
303                 var dopplerCurve = curvesDoplerFrequencyReadingsDictionary
```

```
                    [tupleAntenna];
304                 if (Math.Abs(tag.RfDopplerFrequency) > DOPPLER_FILTER)
305                 {
306                     dopplerCurve.AddPoint(readingTime,
                        tag.RfDopplerFrequency);
307                 }
308                 else
309                 {
310                     //dopplerCurve.AddPoint(readingTime,
                        dopplerCurve.GetCurveLastValue());
311                 }
312
313             }
314             catch (Exception e)
315             {
316                 // Handle .NET errors.
317                 Console.WriteLine("Exception : {0}", e.Message);
318             }
319         }
320
321         /// <summary>
322         /// Setter Ambiente
323         /// </summary>
324         public void SetCurrAmbient(Ambient currAmb)
325         {
326             if (currAmb == null)
327             {
328                 return;
329             }
330             currentAmbient = currAmb;
331         }
332
333         /// <summary>
334         /// Getter Ambiente
335         /// </summary>
336         public Ambient GetCurAmbient()
337         {
338             return currentAmbient;
339         }
340
341         /// <summary>
342         /// Setter Ambiente
343         /// TODO
344         /// </summary>
345         public void SetAmbient(Ambient ambientGuess, int i)
346         {
347             switch (i)
348             {
349                 case 0:
350                     ambPeakTimes = ambientGuess;
351                     break;
352                 case 1:
353                     ambPeakTimeAndMag = ambientGuess;
354                     break;
355                 case 2:
356                     ambLastRSSI = ambientGuess;
```

```
357                     break;
358                 case 3:
359                     ambRSSIMean = ambientGuess;
360                     break;
361                 case 4:
362                     ambRSSIMedian = ambientGuess;
363                     break;
364                 case 5:
365                     ambDopplerEffect = ambientGuess;
366                     break;
367                 case 6:
368                     ambDopplerAndRSSI = ambientGuess;
369                     break;
370             }
371         }
372 
373         /// <summary>
374         /// Getter Ambiente
375         /// TODO
376         /// </summary>
377         public Ambient GetAmbient(int i)
378         {
379             switch (i)
380             {
381                 case 0:
382                     return ambPeakTimes;
383                 case 1:
384                     return ambPeakTimeAndMag;
385                 case 2:
386                     return ambLastRSSI;
387                 case 3:
388                     return ambRSSIMean;
389                 case 4:
390                     return ambRSSIMedian;
391                 case 5:
392                     return ambDopplerEffect;
393                 case 6:
394                     return ambDopplerAndRSSI;
395                 default:
396                     return currentAmbient;
397             }
398         }
399 
400 
401 
402     }
403 }
404 
```

```
1  using System;
2  using System.Collections;
3  using System.Collections.Generic;
4  using System.Linq;
5
6  namespace TG2_RFID
7  {
8
9      public class Curve
10     {
11         /// <summary>
12         /// Holds the curve's name.
13         /// </summary>
14         protected string curveName;
15
16         /// <summary>
17         /// Holds the curve's data separated as X, Y point.
18         /// </summary>
19         protected SortedList<double, double> curveData;
20
21         /// <summary>
22         /// Holds the maximum amount of points in this curve.
23         /// </summary>
24         static protected int maxNumberOfPoints = 12;
25
26         static protected int averageFilterWindow = 3;
27
28         protected double maximumPointX;
29         protected double maximumPointY;
30
31         protected double minimumPointX;
32         protected double minimumPointY;
33
34         public static void PopulateCurveTest(Curve curve)
35         {
36             //curve.AddPoint(0, .5);
37             //curve.AddPoint(0.1, 0.1);
38             //curve.AddPoint(0.2, 0.2);
39             //curve.AddPoint(0.25, 0.25);
40             //curve.AddPoint(0.4, 0.4);
41             //curve.AddPoint(0.5, 0.5);
42             //curve.AddPoint(0.6, 0.6);
43             //curve.AddPoint(0.9, 0.9);
44             //curve.AddPoint(0.99, .95);
45             //curve.AddPoint(.33, .33);
46             //curve.AddPoint(.7, .8);
47             //curve.AddPoint(.75, .8);
48             //curve.AddPoint(.8, .8);
49             //curve.AddPoint(1.3, .6);
50             //curve.AddPoint(1.4, .4);
51             //curve.AddPoint(1.5, .3);
52             //curve.AddPoint(1.6, .2);
53             //curve.AddPoint(1.7, .1);
54             //curve.AddPoint(1.8, -.1);
55             //curve.AddPoint(1.9, -.1);
56             //curve.AddPoint(2.1, -.2);
```

```
57              //curve.AddPoint(2.15, -.3);
58              //curve.AddPoint(2.2, -.4);
59              //curve.AddPoint(2.225, -.5);
60              //curve.AddPoint(2.25, -.6);
61              //curve.AddPoint(2.275, -.7);
62              //curve.AddPoint(2.3, -.8);
63              //curve.AddPoint(2.4, -.9);
64              //curve.AddPoint(2.5, -.95);
65              //curve.AddPoint(2.7, -.95);
66              ////////////////////////////////////
67              ///
68              curve.AddPointWithAvgFilter(-3.14159265358979, 1.39635420686754);
69              curve.AddPointWithAvgFilter(-3.07812613533545, 1.89995829146925);
70              curve.AddPointWithAvgFilter(-3.01465961708111, 1.48854286693781);
71              curve.AddPointWithAvgFilter(-2.95119309882678, 1.0860811022767);
72              curve.AddPointWithAvgFilter(-2.88772658057244, 0.920492037729703);
73              curve.AddPointWithAvgFilter(-2.8242600623181, 1.3838628891516);
74              curve.AddPointWithAvgFilter(-2.76079354406376, 0.933474681122092);
75              curve.AddPointWithAvgFilter(-2.69732702580942, 0.35450756315884);
76              curve.AddPointWithAvgFilter(-2.63386050755508,
                  0.0534229643943726);
77              curve.AddPointWithAvgFilter(-2.57039398930074, 0.838723714241578);
78              curve.AddPointWithAvgFilter(-2.5069274710464, 0.217962442212655);
79              curve.AddPointWithAvgFilter(-2.44346095279206, 0.410322568792965);
80              curve.AddPointWithAvgFilter(-2.37999443453772, 0.33462340878903);
81              curve.AddPointWithAvgFilter(-2.31652791628338,
                  -0.674969032252129);
82              curve.AddPointWithAvgFilter(-2.25306139802904,
                  -0.804442376513671);
83              curve.AddPointWithAvgFilter(-2.1895948797747, -0.997707724013973);
84              curve.AddPointWithAvgFilter(-2.12612836152037, -1.0413203658205);
85              curve.AddPointWithAvgFilter(-2.06266184326603, -1.25870522517678);
86              curve.AddPointWithAvgFilter(-1.99919532501169, -1.35944318507287);
87              curve.AddPointWithAvgFilter(-1.93572880675735, -1.42452521176546);
88              curve.AddPointWithAvgFilter(-1.87226228850301, -1.30958887273105);
89              curve.AddPointWithAvgFilter(-1.80879577024867, -1.15584663542153);
90              curve.AddPointWithAvgFilter(-1.74532925199433, -1.67184080123879);
91              curve.AddPointWithAvgFilter(-1.68186273373999, -1.65681005379094);
92              curve.AddPointWithAvgFilter(-1.61839621548565, -1.64392135469107);
93              curve.AddPointWithAvgFilter(-1.55492969723131, -1.37223601492581);
94              curve.AddPointWithAvgFilter(-1.49146317897697, -1.92358823367669);
95              curve.AddPointWithAvgFilter(-1.42799666072263, -1.19516369166389);
96              curve.AddPointWithAvgFilter(-1.36453014246829, -1.36492208827684);
97              curve.AddPointWithAvgFilter(-1.30106362421395, -1.08783537085571);
98              curve.AddPointWithAvgFilter(-1.23759710595962, -1.26508751441717);
99              curve.AddPointWithAvgFilter(-1.17413058770528,
                  -0.625362236401926);
100             curve.AddPointWithAvgFilter(-1.11066406945094, -0.56684144603304);
101             curve.AddPointWithAvgFilter(-1.0471975511966, -0.856734378351539);
102             curve.AddPointWithAvgFilter(-0.983731032942258,
                  -0.711052078905264);
103             curve.AddPointWithAvgFilter(-0.920264514687919,
                  -0.282403717588411);
104             curve.AddPointWithAvgFilter(-0.85679799643358,
                  -0.645852139012745);
105             curve.AddPointWithAvgFilter(-0.79333147817924, 0.186706066961859);
```

```
106             curve.AddPointWithAvgFilter(-0.729864959924901,
                  -0.466745932262059);
107             curve.AddPointWithAvgFilter(-0.666398441670562,
                  0.176330072978658);
108             curve.AddPointWithAvgFilter(-0.602931923416223,
                  0.0677179747872264);
109             curve.AddPointWithAvgFilter(-0.539465405161884,
                  0.613412607652294);
110             curve.AddPointWithAvgFilter(-0.475998886907544,
                  0.502621087466802);
111             curve.AddPointWithAvgFilter(-0.412532368653205,
                  0.716935659109958);
112             curve.AddPointWithAvgFilter(-0.349065850398866,
                  0.470792234094891);
113             curve.AddPointWithAvgFilter(-0.285599332144526, 1.09232955297835);
114             curve.AddPointWithAvgFilter(-0.222132813890187, 1.41011741790095);
115             curve.AddPointWithAvgFilter(-0.158666295635848, 1.33875190579228);
116             curve.AddPointWithAvgFilter(-0.095199777381509, 1.00794417153841);
117             curve.AddPointWithAvgFilter(-0.03173325912717, 1.91460655432265);
118             curve.AddPointWithAvgFilter(0.03173325912717, 1.7816862048228);
119             curve.AddPointWithAvgFilter(0.095199777381509, 1.37592051147737);
120             curve.AddPointWithAvgFilter(0.158666295635848, 1.37860742057852);
121             curve.AddPointWithAvgFilter(0.222132813890187, 1.2872818200005);
122             curve.AddPointWithAvgFilter(0.285599332144526, 1.98959707400003);
123             curve.AddPointWithAvgFilter(0.349065850398866, 1.18834528115876);
124             curve.AddPointWithAvgFilter(0.412532368653205, 1.99317242166135);
125             curve.AddPointWithAvgFilter(0.475998886907544, 1.60381272234308);
126             curve.AddPointWithAvgFilter(0.539465405161884, 1.80561808342577);
127             curve.AddPointWithAvgFilter(0.602931923416223, 1.72570961909859);
128             curve.AddPointWithAvgFilter(0.666398441670562, 1.6386525947751);
129             curve.AddPointWithAvgFilter(0.729864959924901, 1.75586085557358);
130             curve.AddPointWithAvgFilter(0.79333147817924, 1.49256828391793);
131             curve.AddPointWithAvgFilter(0.85679799643358, 1.12734384227661);
132             curve.AddPointWithAvgFilter(0.920264514687919, 0.719248621295077);
133             curve.AddPointWithAvgFilter(0.983731032942258, 0.65975286277186);
134             curve.AddPointWithAvgFilter(1.0471975511966, 0.591019662713436);
135             curve.AddPointWithAvgFilter(1.11066406945094, 1.28613295844699);
136             curve.AddPointWithAvgFilter(1.17413058770528, 1.12065310853309);
137             curve.AddPointWithAvgFilter(1.23759710595962, 1.00736033247674);
138             curve.AddPointWithAvgFilter(1.30106362421395, 0.624858768074646);
139             curve.AddPointWithAvgFilter(1.36453014246829, 0.0810994893070038);
140             curve.AddPointWithAvgFilter(1.42799666072263, 0.0881309104803278);
141             curve.AddPointWithAvgFilter(1.49146317897697, 0.0115093591309446);
142             curve.AddPointWithAvgFilter(1.55492969723131, 0.472983281472946);
143             curve.AddPointWithAvgFilter(1.61839621548565, 0.171282500319029);
144             curve.AddPointWithAvgFilter(1.68186273373999, 0.789575346431925);
145             curve.AddPointWithAvgFilter(1.74532925199433, 0.0455145603400934);
146             curve.AddPointWithAvgFilter(1.80879577024867, 0.414848567793334);
147             curve.AddPointWithAvgFilter(1.87226228850301, 0.976707108463735);
148             curve.AddPointWithAvgFilter(1.93572880675735, 0.409173496221809);
149             curve.AddPointWithAvgFilter(1.99919532501169, 0.461345089757619);
150             curve.AddPointWithAvgFilter(2.06266184326603, 1.2881157063007);
151             curve.AddPointWithAvgFilter(2.12612836152037, 1.15891840436669);
152             curve.AddPointWithAvgFilter(2.1895948797747, 0.953258370681676);
153             curve.AddPointWithAvgFilter(2.25306139802904, 0.879173622417986);
154             curve.AddPointWithAvgFilter(2.31652791628338, 1.48330907109987);
```

```
155             curve.AddPointWithAvgFilter(2.37999443453772, 1.21653245695899);
156             curve.AddPointWithAvgFilter(2.44346095279206, 1.7273615984108);
157             curve.AddPointWithAvgFilter(2.5069274710464, 1.8059154068297);
158             curve.AddPointWithAvgFilter(2.57039398930074, 1.91125743207817);
159             curve.AddPointWithAvgFilter(2.63386050755508, 1.89979940409336);
160             curve.AddPointWithAvgFilter(2.69732702580942, 1.21360887389898);
161             curve.AddPointWithAvgFilter(2.76079354406376, 1.47324949905646);
162             curve.AddPointWithAvgFilter(2.8242600623181, 2.09487483004092);
163             curve.AddPointWithAvgFilter(2.88772658057244, 2.11566891751715);
164             curve.AddPointWithAvgFilter(2.95119309882678, 1.34972184058067);
165             curve.AddPointWithAvgFilter(3.01465961708111, 1.31136365213583);
166             curve.AddPointWithAvgFilter(3.07812613533545, 1.68187849713421);
167             curve.AddPointWithAvgFilter(3.14159265358979, 1.15568185651978);
168         }
169
170         /// <summary>
171         /// Adds a point for the curve.
172         /// </summary>
173         /// <param name="x">The x coordinate.</param>
174         /// <param name="y">The y coordinate.</param>
175         public void AddPoint(double x, double y)
176         {
177             //var watch = System.Diagnostics.Stopwatch.StartNew();
178             try
179             {
180                 curveData.Add(x, y);
181                 if (maximumPointY < y)
182                 {
183                     maximumPointY = y;
184                     maximumPointX = x;
185                 }
186                 if (minimumPointY > y)
187                 {
188                     minimumPointY = y;
189                     minimumPointX = x;
190                 }
191                 while (curveData.Count >= maxNumberOfPoints)
192                 {
193                     var removedX = curveData.Keys[0];
194                     curveData.Remove(curveData.Keys[0]);
195                     if (Double.Equals(removedX, minimumPointX))
196                     {
197                         var XY = CalculateCurveMinPoint();
198                         minimumPointX = XY.Item1;
199                         minimumPointY = XY.Item2;
200                     }
201                     if (Double.Equals(removedX, maximumPointX))
202                     {
203                         var XY = CalculateCurveMaxPoint();
204                         maximumPointX = XY.Item1;
205                         maximumPointY = XY.Item2;
206                     }
207                 }
208             }
209             catch(Exception e)
210             {
```

```
211                 // Handle .NET errors.
212                 Console.WriteLine("Exception : {0}", e.Message);
213             }
214             //watch.Stop();
215             //var elapsedMs = watch.ElapsedMilliseconds;
216             //Console.WriteLine("AddPointTIme in ms : {0}", elapsedMs);
217         }
218 
219         public void AddPointWithAvgFilter(double x, double y)
220         {
221             if (curveData.Count >= averageFilterWindow)
222             {
223                 var avg = y + curveData.Values[curveData.Count - 1] +
                      curveData.Values[curveData.Count - (averageFilterWindow -
                      1)];
224                 avg /= averageFilterWindow;
225                 AddPoint(x, avg);
226             }
227             else
228             {
229                 AddPoint(x, y);
230             }
231         }
232 
233         /// <summary>
234         /// Prints the curve in console.
235         /// </summary>
236         public void PrintCurveInConsole()
237         {
238             const char BLANK = ' ';
239             const char DOT = '.';
240             const char X = 'x';
241             const int cMaxLineChars = 79;
242             const int cHalf = cMaxLineChars / 2;
243             char[] LINE = new char[cMaxLineChars];
244 
245             for (int i = 0; i < LINE.Length; i++)
246             {
247                 LINE[i] = DOT;
248             }
249             Console.WriteLine(LINE);
250             for (int i = 0; i < LINE.Length; i++)
251             {
252                 LINE[i] = BLANK;
253             }
254 
255             int loc;
256             //var maxY = getCurveMaxAbsY();
257             var maxY = 2.2;
258             LINE[cHalf] = DOT;
259             foreach (var pair in curveData)
260             {
261                 loc = (int)Math.Round(cMaxLineChars * (pair.Value + maxY) / (2
                        * maxY));
262                 //loc = (int)Math.Round(cMaxLineChars * (pair.Value + maxY) /
                      (2 * maxY));
```

```
263                 if (loc == LINE.Length)
264                 {
265                     LINE[loc - 1] = X;
266                 } else
267                 {
268                     LINE[loc] = X;
269                 }
270                 Console.WriteLine(LINE);
271                 for (int i = 0; i < LINE.Length; i++)
272                 {
273                     LINE[i] = BLANK;
274                 }
275                 LINE[cHalf] = DOT;
276             }
277         }
278 
279         public void PrintCurveLastValue()
280         {
281             Console.WriteLine(curveData[curveData.Keys[0]]);
282         }
283 
284         public double GetCurveLastValue()
285         {
286             if (curveData.Count != 0)
287             {
288                 var x = curveData.Keys[curveData.Keys.Count - 1];
289                 return curveData[x];
290             }
291             else
292             {
293                 return 0;
294             }
295 
296         }
297 
298         /*!
299          * Writes info about cardholder: name, EPC, Ambient and time-entered-
              ambient
300          */
301         public void WriteDataToFile()
302         {
303 
304         }
305 
306         /// <summary>
307         /// Getter the minimum value for x coordinate.
308         /// </summary>
309         /// <returns>The curve minimum x.</returns>
310         public double GetCurveMinX()
311         {
312             return curveData.Keys[0];
313         }
314 
315         /// <summary>
316         /// Getter for the index's value for Y
317         /// </summary>
```

```
318         public double GetCurveIndexY(int i)
319         {
320             if (curveData.Values.Count() > i)
321                 return curveData.Values[i];
322             else
323                 return 0;
324         }
325 
326         /// <summary>
327         /// Getter for the index's value for X
328         /// </summary>
329         public double GetCurveIndexX(int i)
330         {
331             if (curveData.Values.Count() > i)
332                 return curveData.Keys[i];
333             else
334                 return 0;
335         }
336 
337         /// <summary>
338         /// Getter the maximum value for x coordinate.
339         /// </summary>
340         public double GetCurveMaxX()
341         {
342             return curveData.Keys[curveData.Count - 1];
343         }
344 
345         /// <summary>
346         /// Getter the curve coordinate for the y maximum value.
347         /// </summary>
348         public Tuple<double, double> CalculateCurveMaxPoint()
349         {
350             maximumPointX = Double.NegativeInfinity;
351             maximumPointY = Double.NegativeInfinity;
352             if (curveData.Count != 0)
353             {
354                 foreach (var pair in curveData)
355                 {
356                     if (pair.Value > maximumPointY)
357                     {
358                         maximumPointX = pair.Key;
359                         maximumPointY = pair.Value;
360                     }
361                 }
362             }
363 
364             return Tuple.Create<double, double>(maximumPointX, maximumPointY);
365         }
366 
367         public Tuple<double, double> GetCurveMaxPoint()
368         {
369             return Tuple.Create<double, double>(maximumPointX, maximumPointY);
370         }
371 
372         /// <summary>
373         /// Getter the curve coordinate for the y minimum value.
```

```
374         /// </summary>
375         public Tuple<double, double> CalculateCurveMinPoint()
376         {
377             minimumPointX = Double.PositiveInfinity;
378             minimumPointY = Double.PositiveInfinity;
379             foreach (var pair in curveData)
380             {
381                 if (pair.Value < minimumPointY)
382                 {
383                     minimumPointX = pair.Key;
384                     minimumPointY = pair.Value;
385                 }
386             }
387             return Tuple.Create<double, double>(minimumPointX, minimumPointY);
388         }
389 
390         public Tuple<double, double> GetCurveMinPoint()
391         {
392             return Tuple.Create<double, double>(minimumPointX, minimumPointY);
393         }
394 
395         /// <summary>
396         /// Getter the curve median y coordinates.
397         /// </summary>
398         public double GetMedianY()
399         {
400             double medianY = 0;
401             if (curveData.Count > 2)
402             {
403                 var values = curveData.Values;
404                 List<double> list = new List<double>(values);
405                 list.Sort();
406                 if (list.Count % 2 != 0)
407                 {
408                     medianY = list[list.Count / 2];
409                 }
410                 else
411                 {
412                     medianY = list[1 + list.Count / 2];
413                 }
414             }
415             return medianY;
416         }
417 
418         /// <summary>
419         /// Getter the curve mean y coordinates.
420         /// </summary>
421         public double CalculateMeanY()
422         {
423             double sumY = 0;
424             double medianY = 0;
425             foreach (var pair in curveData)
426             {
427                 sumY += pair.Value;
428             }
429             if (curveData.Count != 0)
```

```
430             {
431                 medianY = sumY / curveData.Count;
432             }
433             return medianY;
434         }
435 
436         /// <summary>
437         /// Getter for the crossing treshold between positive and negative
              values.
438         /// </summary>
439         public Tuple<double, double> CalculateCrossingPoint()
440         {
441             //var watch = System.Diagnostics.Stopwatch.StartNew();
442             double crossingPointX = Double.NaN;
443             double crossingPointY = Double.NaN;
444             bool isFirstSignPositive = false;
445             for(int i = curveData.Count - 1; i >= 0; i--)
446             {
447                 var x = curveData.Keys[i];
448                 var y = curveData[x];
449                 if (i == curveData.Count - 1)
450                 {
451                     isFirstSignPositive = y > 0 ? true : false;
452                     continue;
453                 }
454                 //if ((y < 0 && isFirstSignPositive) || (y > 0 && !
                      isFirstSignPositive))
455                 if ((y > 0 && !isFirstSignPositive))
456                 {
457                     if (i < curveData.Count - 2)
458                     {
459                         x += curveData.Keys[i + 1];
460                         x *= 0.5;
461                         y += curveData[curveData.Keys[i + 1]];
462                         y *= 0.5;
463                     }
464                     crossingPointY = x;
465                     crossingPointX = y;
466                     break;
467                 }
468             }
469             //watch.Stop();
470             //var elapsedMs = watch.ElapsedMilliseconds;
471             //Console.WriteLine("CrossingPoint in ms : {0}", elapsedMs);
472             return Tuple.Create(crossingPointX, crossingPointY);
473         }
474 
475         /// <summary>
476         /// Constructor for a curve.
477         /// </summary>
478         public Curve()
479         {
480             curveData = new SortedList<double, double>();
481             maximumPointX = Double.NegativeInfinity;
482             maximumPointY = Double.NegativeInfinity;
483             minimumPointX = Double.PositiveInfinity;
```

```
484             minimumPointY = Double.PositiveInfinity;
485         }
486
487         public int GetSize()
488         {
489             return curveData.Count;
490         }
491
492         public IList<Tuple<double, double>> CalculatePeaks()
493         {
494             //var watch = System.Diagnostics.Stopwatch.StartNew();
495             IList<double> values = curveData.Values;
496             int rangeOfPeaks = (int)(Math.Round(.3 * values.Count));
497
498             List<Tuple<double, double>> peaks = new List<Tuple<double,
                  double>>();
499
500             double current;
501             IEnumerable<double> range;
502
503             int checksOnEachSide = rangeOfPeaks / 2;
504             if (values.Count == 0)
505             {
506                 return peaks;
507             }
508             else if (values.Count < checksOnEachSide)
509             {
510                 peaks.Add(Tuple.Create(maximumPointX, maximumPointY));
511             }
512             else
513             {
514                 for (int i = 0; i < values.Count; i++)
515                 {
516                     current = values[i];
517                     range = values;
518
519                     if (i > checksOnEachSide)
520                     {
521                         range = range.Skip(i - checksOnEachSide);
522                     }
523
524                     range = range.Take(rangeOfPeaks);
525                     if ((range.Count() > 0) && (current == range.Max()))
526                     {
527                         peaks.Add(Tuple.Create(curveData.Keys[i],
                        curveData.Values[i]));
528                     }
529                 }
530             }
531             //watch.Stop();
532             //var elapsedMs = watch.ElapsedMilliseconds;
533             //Console.WriteLine("PeaksPoint in ms : {0}", elapsedMs);
534             return peaks;
535         }
536
537         public bool CompareCurveMaximums(Curve otherCurve)
```

```
538         {
539             return GetCurveMaxPoint().Item2 > otherCurve.GetCurveMaxPoint
                ().Item2;
540         }
541
542         public bool CompareCurveLastValues(Curve otherCurve)
543         {
544             return GetCurveLastValue() > otherCurve.GetCurveLastValue();
545         }
546
547         public bool CompareCurveMeans(Curve otherCurve)
548         {
549             return CalculateMeanY() >= otherCurve.CalculateMeanY();
550         }
551
552         public bool CompareCurveMedians(Curve otherCurve)
553         {
554             return GetMedianY() >= otherCurve.GetMedianY();
555         }
556
557         public bool CompareCurveLastPeak(Curve otherCurve)
558         {
559             var peakListLast = CalculatePeaks();
560             var peakListOther = otherCurve.CalculatePeaks();
561
562             return (peakListLast.Count > 0 && peakListOther.Count > 0 &&
                (peakListLast[peakListLast.Count - 1].Item1 > peakListOther
                [peakListOther.Count - 1].Item1))
563                    || (peakListLast.Count == 0 || peakListOther.Count == 0 &&
                    CompareCurveMaximums(otherCurve));
564         }
565     }
566 }
567
568
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4  using System.IO;
5
6  namespace TG2_RFID
7  {
8      public class FileHandler
9      {
10
11         protected string filePath;
12
13         public void SetFileHandler()
14         {
15             filePath = @Environment.GetFolderPath
                 (Environment.SpecialFolder.Desktop) + "\\log_" +
                 DateTime.Now.ToString("yyyyMMdd_HH-mm-ss") + ".csv";
16
17         }
18
19         public void WriteToFile(Cardholder person, string epc, string reader,
             ushort ant)
20         {
21             // Set File parameters
22             string csvSeperator = ";";
23             StringBuilder streamOutput = new StringBuilder();
24
25             var antenna = Tuple.Create<string,ushort> (reader,ant);
26             var strRSSIcurve = person.GetPowerCurve(antenna).GetCurveIndexY
                 (0).ToString() + ";" +
27                                person.GetPowerCurve(antenna).GetCurveIndexY
                       (1).ToString() + ";" +
28                                person.GetPowerCurve(antenna).GetCurveIndexY
                       (2).ToString() + ";" +
29                                person.GetPowerCurve(antenna).GetCurveIndexY
                       (3).ToString() + ";" +
30                                person.GetPowerCurve(antenna).GetCurveIndexY
                       (4).ToString() + ";" +
31                                person.GetPowerCurve(antenna).GetCurveIndexY
                       (5).ToString() + ";" +
32                                person.GetPowerCurve(antenna).GetCurveIndexY
                       (6).ToString() + ";" +
33                                person.GetPowerCurve(antenna).GetCurveIndexY
                       (7).ToString() + ";" +
34                                person.GetPowerCurve(antenna).GetCurveIndexY
                       (8).ToString() + ";" +
35                                person.GetPowerCurve(antenna).GetCurveIndexY
                       (9).ToString() + ";" +
36                                person.GetPowerCurve(antenna).GetCurveIndexY
                       (10).ToString() + ";"
37                                ;
38
39             var strRSSItime =  person.GetPowerCurve(antenna).GetCurveIndexX
                 (0).ToString() + ";" +
40                                person.GetPowerCurve(antenna).GetCurveIndexX
                       (1).ToString() + ";" +
```

```
41                                  person.GetPowerCurve(antenna).GetCurveIndexX
                        (2).ToString() + ";" +
42                                  person.GetPowerCurve(antenna).GetCurveIndexX
                        (3).ToString() + ";" +
43                                  person.GetPowerCurve(antenna).GetCurveIndexX
                        (4).ToString() + ";" +
44                                  person.GetPowerCurve(antenna).GetCurveIndexX
                        (5).ToString() + ";" +
45                                  person.GetPowerCurve(antenna).GetCurveIndexX
                        (6).ToString() + ";" +
46                                  person.GetPowerCurve(antenna).GetCurveIndexX
                        (7).ToString() + ";" +
47                                  person.GetPowerCurve(antenna).GetCurveIndexX
                        (8).ToString() + ";" +
48                                  person.GetPowerCurve(antenna).GetCurveIndexX
                        (9).ToString() + ";" +
49                                  person.GetPowerCurve(antenna).GetCurveIndexX
                        (10).ToString() + ";"
50                                  ;
51
52                  var strDopplercurve = person.GetDopplerEffectCurve
                      (antenna).GetCurveIndexY(0).ToString() + ";" +
53                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexY(1).ToString() + ";" +
54                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexY(2).ToString() + ";" +
55                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexY(3).ToString() + ";" +
56                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexY(4).ToString() + ";" +
57                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexY(5).ToString() + ";" +
58                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexY(6).ToString() + ";" +
59                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexY(7).ToString() + ";" +
60                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexY(8).ToString() + ";" +
61                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexY(9).ToString() + ";" +
62                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexY(10).ToString()
63                                        ;
64
65                  var strDopplertime =  person.GetDopplerEffectCurve
                      (antenna).GetCurveIndexX(0).ToString() + ";" +
66                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexX(1).ToString() + ";" +
67                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexX(2).ToString() + ";" +
68                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexX(3).ToString() + ";" +
69                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexX(4).ToString() + ";" +
70                                        person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexX(5).ToString() + ";" +
```

```
71                                  person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexX(6).ToString() + ";" +
72                                  person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexX(7).ToString() + ";" +
73                                  person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexX(8).ToString() + ";" +
74                                  person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexX(9).ToString() + ";" +
75                                  person.GetDopplerEffectCurve
                        (antenna).GetCurveIndexX(10).ToString()
76                                  ;
77
78              string[][] dataOutput = new string[][]{
79                                    new string[]{person.GetName(), epc,
                        reader, ant.ToString(), DateTime.Now.ToString("dd/MM/yyyy;
                         HH:mm:ss:ffff"), person.GetCurAmbient().GetName(),";",
                        strRSSIcurve, strRSSItime, strDopplercurve,";",
                        strDopplertime, ";", person.GetAmbient(0).GetName(),
                        person.GetAmbient(2).GetName(), person.GetAmbient
                        (3).GetName(), person.GetAmbient(4).GetName(),
                        person.GetAmbient(5).GetName(), person.GetAmbient
                        (6).GetName() }
80                                  };
81              int length = dataOutput.GetLength(0);
82              for (int i = 0; i < length; i++)
83                  streamOutput.AppendLine(string.Join(csvSeperator, dataOutput
                      [i]));
84
85              // Appends more lines to the csv file
86              File.AppendAllText(filePath, streamOutput.ToString());
87          }
88
89          public void CreateFile()
90          {
91              // Set File parameters
92              string csvSeperator = ";";
93              StringBuilder streamOutput = new StringBuilder();
94
95              string[][] dataOutput = new string[][]{
96                                    new string[]{"Nome", "EPC", "Leitora",
                        "Antena", "Data", "Horario", "Ambiente Correto", ";",
                        "Curva RSSI", ";;;;;;;;;",";", "Tempo RSSI", ";;;;;;;;;",
                        ";", "Curva Doppler", ";;;;;;;;;", ";", "Tempo Doppler",
                        ";;;;;;;;;",  ";;", "Ambiente-LAST-RSSI-PEAK-TIME",
                        "Ambiente-RSSI-Last-value", "Ambiente-RSSI-Mean",
                        "Ambiente-RSSI-Median", "Ambiente-Doppler" ,
                        "DopplerAndRSSI"}
97                                    };
98              int length = dataOutput.GetLength(0);
99              for (int i = 0; i < length; i++)
100                 streamOutput.AppendLine(string.Join(csvSeperator, dataOutput
                      [i]));
101
102             // Create and write the csv file
103             File.WriteAllText(filePath, streamOutput.ToString());
104         }
```

```
105
106
107         }
108     }
109
```

```
1  /*
2   * Trabalho de Graduação II - Controle antecipativo de sistemas de conforto
      térmico usando RFID
3   *
4   * Autor: André Abreu Rodrigues de Almeida
5   *        Graduando em Engenharia Mecatrônica
6   *        Matrícula: 12/0007100
7   * Departamento de Engenharia Elétrica
8   * Universidade de Brasília - UnB
9   *
10  *
11  * Brasília, Agosto a Dezembro de 2019
12  *
13  */
14
15 /* Aviso: A biblioteca 'Octane SDK' foi desenvolvida e é propriedade da
      empresa IMPINJ, INC., e está sendo utilizada com base na licença de software
      aberto descrita a seguir
16  *
17  * Para mais informações, visitar o site: https://support.impinj.com/hc/en-us/
      articles/360000468370-Software-Tools-License-Disclaimer
18  */
19
20
21 /*PLEASE READ THE FOLLOWING LICENSE & DISCLAIMER (“AGREEMENT”) CAREFULLY
     BEFORE USING ANY SOFTWARE TOOLS (AS DEFINED BELOW) MADE AVAILABLE TO YOU
     (“LICENSEE”) BY IMPINJ, INC. (“IMPINJ”). BY USING THE SOFTWARE TOOLS, YOU
     ACKNOWLEDGE THAT YOU HAVE READ AND UNDERSTOOD ALL THE TERMS AND CONDITIONS
     OF THE AGREEMENT, YOU WILL BE CONSENTING TO BE BOUND BY THEM, AND YOU ARE
     AUTHORIZED TO DO SO. IF YOU DO NOT ACCEPT THESE TERMS AND CONDITIONS, DO NOT
      USE THE SOFTWARE TOOLS.
22
23 1. PURPOSE OF AGREEMENT. From time to time, Impinj technical personnel may
     make available to Licensee certain software, including code (in source and
     object form), tools, libraries, configuration files, translations, and
     related documentation (collectively, “Software Tools”), upon specific
     request or to assist with a specific deployment. This Agreement sets forth
     Licensee's limited rights and Impinj's limited obligations with respect to
     the Software Tools. Licensee acknowledges that Impinj provides the Software
     Tools free of charge. This Agreement does not grant any rights with respect
     to Impinj standalone software products (e.g., ItemSense, ItemEncode,
     SpeedwayConnect) or the firmware on Impinj hardware, all of which are
     subject to separate license terms.
24
25 2. LIMITED LICENSE. Subject to the terms and conditions of this Agreement,
     Impinj hereby grants to Licensee a limited, royalty-free, worldwide, non-
     exclusive, perpetual and irrevocable (except as set forth below), non-
     transferable license, without right of sublicense, to (a) use the Software
     Tools and (b) only with respect to Software Tools provided in source code
     form, modify and create derivative works of such Software Tools, in each
     case, solely for Licensee’s internal development related to the deployment
     of Impinj products (“Purpose”). The Software Tools may only be used by
     employees of Licensee that must have access to the Software Tools in
     connection with the Purpose.
26
27 3. TERMINATION. Impinj may immediately terminate this Agreement if Licensee
```

```
      breaches any provision hereof. Upon the termination of this Agreement,
      Licensee must (a) discontinue all use of the Software Tools, (b) uninstall
      the Software Tools from its systems, (c) destroy or return to Impinj all
      copies of the Software Tools and any other materials provided by Impinj, and
       (d) promptly provide Impinj with written confirmation (including via email)
       of Licensee's compliance with these provisions. Sections 4-10 will survive
      termination of this Agreement.
28
29    4. OWNERSHIP. The Software Tools are licensed, not sold, by Impinj to
      Licensee. Impinj and its suppliers own and retain all right, title, and
      interest, including all intellectual property rights, in and to the Software
       Tools. Except for those rights expressly granted in this Agreement, no
      other rights are granted, either express or implied, to Licensee. Impinj
      reserves the right to develop, price and sell software products that have
      features similar to or competitive with Software Tools. Licensee grants
      Impinj a limited, royalty-free, worldwide, perpetual and irrevocable,
      transferable, sublicensable, license to Licensee's derivative works of
      Software Tools; provided that Licensee has no obligation under this
      Agreement to deliver to Impinj any such derivative works.
30
31    5. CONFIDENTIALITY. In order to protect the trade secrets and proprietary
      know-how contained in the Software Tools, Licensee will not decompile,
      disassemble, or reverse engineer, or otherwise attempt to gain access to the
       source code or algorithms of the Software Tools (unless Impinj provides the
       Software Tools in source code format). Licensee will maintain the
      confidentiality of and not disclose to any third party: (a) all non-public
      information disclosed by Impinj to Licensee under this Agreement and (b) all
       performance data and all other information obtained through the Software
      Tools.
32
33    6. WARRANTY DISCLAIMER. LICENSEE AC KNOWLEDGES THAT IMPINJ PROVIDES THE
      SOFTWARE TOOLS FREE OF CHARGE AND ONLY FOR THE PURPOSE. ACCORDINGLY, THE
      SOFTWARE TOOLS ARE PROVIDED "AS IS" WITHOUT QUALITY CHECK, AND IMPINJ DOES
      NOT WARRANT THAT THE SOFTWARE TOOLS WILL OPERATE WITHOUT ERROR OR
      INTERRUPTION OR MEET ANY PERFORMANCE STANDARD OR OTHER EXPECTATION. IMPINJ
      EXPRESSLY DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING THE
      IMPLIED WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT, QUALITY, ACCURACY,
      AND FITNESS FOR A PARTICULAR PURPOSE. IMPINJ IS NOT OBLIGATED IN ANY WAY TO
      PROVIDE SUPPORT OR OTHER MAINTENANCE WITH RESPECT TO THE SOFTWARE TOOLS.
34
35    7. LIMITATION OF LIABILITY. THE TOTAL LIABILITY OF IMPINJ ARISING OUT OF OR
      RELATED TO THE SOFTWARE TOOLS WILL NOT EXCEED THE TOTAL AMOUNT PAID BY
      LICENSEE TO IMPINJ PURSUANT TO THIS AGREEMENT. IN NO EVENT WILL IMPINJ HAVE
      LIABILITY FOR ANY INDIRECT, INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES,
      EVEN IF ADVISED OF THE POSSIBILITY OF THESE DAMAGES. THESE LIMITATIONS WILL
      APPLY NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY LIMITED REMEDY
       IN THIS AGREEMENT.
36
37    8. THIRD PARTY SOFTWARE. The Software Tools may contain software created by a
      third party. Licensee's use of any such third party software is subject to
      the applicable license terms and this Agreement does not alter those license
       terms. Licensee may not subject any portion of the Software Tools to an
      open source license.
38
39    9. RESTRICTED USE. Licensee will comply with all applicable laws and
      regulations to preclude the acquisition by any governmental agency of
```

```
      unlimited rights to technical data, software, and documentation provided
      with Software Tools, and include the appropriate “Restricted Rights” or
      “Limited Rights” notices required by the applicable U.S. or foreign
      government agencies. Licensee will comply in all respects with all U.S. and
      foreign export and re-export laws and regulations applicable to the
      technology and documentation provided hereunder.
40
41  10. MISCELLANEOUS. This Agreement will be governed by the laws of the State of
       Washington, U.S.A without reference to conflict of law principles. All
      disputes arising out of or related to it, will be subject to the exclusive
      jurisdiction of the state and federal courts located in King County,
      Washington, and the parties agree and submit to the personal and exclusive
      jurisdiction and venue of these courts. Licensee will not assign this
      Agreement, directly or indirectly, by operation of law or otherwise, without
       the prior written consent of Impinj. This Agreement (and any applicable
      nondisclosure agreement) is the entire agreement between the parties
      relating to the Software Tools. No waiver or modification of this Agreement
      will be valid unless contained in a writing signed by each party.
42  */
43
44  #define DEBUG
45
46  using System;
47  using Impinj.OctaneSdk;
48  using System.Collections;
49  using System.Collections.Generic;
50  using System.IO;
51  using System.Text;
52  using System.Threading;
53
54  namespace TG2_RFID
55  {
56      class GlobalData
57      {
58          public static int RSSILowPassFilter = -60;
59
60          public static FileHandler filehandler = new FileHandler();
61      }
62
63      class Program
64      {
65          // Create a collection to hold all the ImpinjReader instances.
66          protected static List<ImpinjReader> readers = new List<ImpinjReader>
              ();
67
68          static void Main(/*string[] args*/)
69          {
70              //Console.WriteLine("Test Step");
71              //Curve cc = new Curve();
72              //Curve.PopulateCurveTest(cc);
73              //cc.PrintCurveInConsole();
74              //Console.WriteLine("MeanY:{0}", cc.CalculateMeanY());
75              //Console.WriteLine("MedianY:{0}", cc.GetMedianY());
76              //var maxPoint = cc.GetCurveMaxPoint();
77              //var minPoint = cc.GetCurveMinPoint();
78              //Console.WriteLine("MaxPoint<{0}, {1}>", maxPoint.Item1,
```

```
                maxPoint.Item2);
79              //Console.WriteLine("MinPoint<{0}, {1}>", minPoint.Item1,
                minPoint.Item2);
80              //Console.WriteLine("MinX:{0}, MaxX:{1}", cc.GetCurveMinX(),
                cc.GetCurveMaxX());
81              //var crossingPoint = cc.CalculateCrossingPoint();
82              //Console.WriteLine("CrossingPoint<{0}, {1}>",
                crossingPoint.Item1, crossingPoint.Item2);
83              //var peaks = cc.CalculatePeaks();
84              //Console.WriteLine("NPeaks: {0}", peaks.Count);
85              //foreach (var peak in peaks)
86              //{
87              //    Console.WriteLine("Peak: <{0},{1}>", peak.Item1,
                peak.Item2);
88              //}
89
90
91              //Console.WriteLine("End Test Step");
92              //return;
93
94              try
95              {
96                  // Sets the output file path
97                  GlobalData.filehandler.SetFileHandler();
98                  GlobalData.filehandler.CreateFile();
99
100                 // holds the paths to the readers
101                 string hostname1 = "speedwayr-10-9f-3f.local";
102                 string hostname2 = "speedwayr-10-9f-c8.local";
103                 string hostname3 = "speedwayr-10-9f-bb.local";
104
105                 // Create two reader instances and add them to the List of
                    readers.
106                 readers.Add(new ImpinjReader(hostname1, "Reader #1"));
107                 readers.Add(new ImpinjReader(hostname2, "Reader #2"));
108                 readers.Add(new ImpinjReader(hostname3, "Reader #3"));
109
110                 //Create map of rooms
111                 Project.RegisterNewAmbient(0, new Ambient("Area_Externa(0)"));
112                 Project.RegisterNewAmbient(1, new Ambient("Sala_Principal
                    (1)"));
113                 Project.RegisterNewAmbient(2, new Ambient("Sala_Reuniao(2)"));
114                 Project.RegisterNewAmbient(3, new Ambient("Corredor_Baias
                    (3)"));
115
116                 //Create map of transitions
117                 Transition transition1 = new Transition
                      (Project.GetAmbientInstance(0), "Reader #1", 1,
                      Project.GetAmbientInstance(1), "Reader #1", 2);
118                 Transition transition2 = new Transition
                      (Project.GetAmbientInstance(1), "Reader #2", 2,
                      Project.GetAmbientInstance(2), "Reader #2", 1);
119                 Transition transition3 = new Transition
                      (Project.GetAmbientInstance(1), "Reader #3", 1,
                      Project.GetAmbientInstance(3), "Reader #3", 2);
120                 Project.RegisterNewTransition(Tuple.Create<string, ushort>
```

```
                    ("Reader #1", 1), transition1);
121                 Project.RegisterNewTransition(Tuple.Create<string, ushort>
                    ("Reader #1", 2), transition1);
122                 Project.RegisterNewTransition(Tuple.Create<string, ushort>
                    ("Reader #2", 1), transition2);
123                 Project.RegisterNewTransition(Tuple.Create<string, ushort>
                    ("Reader #2", 2), transition2);
124                 Project.RegisterNewTransition(Tuple.Create<string, ushort>
                    ("Reader #3", 1), transition3);
125                 Project.RegisterNewTransition(Tuple.Create<string, ushort>
                    ("Reader #3", 2), transition3);
126
127                 //Create Map of Cardholders
128                 Project.PopulateProjectCardholders();
129
130                 // Loop through the List of readers to configure and start
                    them.
131                 foreach (ImpinjReader reader in readers)
132                 {
133                     // Connect to the reader
134                     reader.Connect();
135
136                     // Get the default settings
137                     // We'll use these as a starting point
138                     // and then modify the settings we're
139                     // interested in.
140                     Settings settings = reader.QueryDefaultSettings();
141
142                     settings.Report.IncludeAntennaPortNumber = true;
143                     settings.Report.IncludeFirstSeenTime = true;
144                     settings.Report.IncludeLastSeenTime = true;
145                     settings.Report.IncludeSeenCount = true;
146                     settings.Report.IncludeDopplerFrequency = true;
147                     settings.Report.IncludePeakRssi = true;
148                     // Send a tag report for every tag read
149                     settings.Report.Mode = ReportMode.Individual;//
                    BatchAfterStop;
150
151                     // Reading tags for 1 seconds every 0.25 second
152                     //settings.AutoStart.Mode = AutoStartMode.Periodic;
153                     //settings.AutoStart.PeriodInMs = 500;
154                     //settings.AutoStop.Mode = AutoStopMode.Duration;
155                     //settings.AutoStop.DurationInMs = 500;
156
157                     //Settings de antena
158                     settings.Antennas.DisableAll();
159                     settings.Antennas.GetAntenna(1).IsEnabled = true;
160                     settings.Antennas.GetAntenna(2).IsEnabled = true;
161                     // Set all the antennas to the max transmit power and
                    receive sensitivity
162                     settings.Antennas.TxPowerMax = true;
163                     settings.Antennas.RxSensitivityMax = true;
164                     // Or set all antennas to a specific value in dBm
165                     //settings.Antennas.TxPowerInDbm = 28.0;
166                     //settings.Antennas.RxSensitivityInDbm = -70.0;
167                     // Or set each antenna individually
```

```
168                     //settings.Antennas.GetAntenna(1).MaxTxPower = true;
169                     //settings.Antennas.GetAntenna(1).MaxRxSensitivity = true;
170                     //settings.Antennas.GetAntenna(2).TxPowerInDbm = 30.0;
171                     //settings.Antennas.GetAntenna(2).RxSensitivityInDbm =
                        -70.0;
172                     // ...
173
174
175                     settings.ReaderMode = ReaderMode.DenseReaderM8;
176
177
178
179                     // Apply the newly modified settings.
180                     reader.ApplySettings(settings);
181
182                     // Assign the TagsReported event handler.
183                     // This specifies which method to call
184                     // when tags reports are available.
185                     reader.TagsReported += Captura_tags;
186
187                     // Start reading.
188                     reader.Start();
189                 }
190
191                 // Wait for the user to press enter.
192                 //Console.WriteLine("Press enter to exit.");
193                 //Console.ReadKey();
194                 while(true)
195                 {
196                     Console.WriteLine("Press 0, 1, 2, and 3 to set debuggin
                        cardholder ambient and anything else to exit.");
197                     var stringConsole = Console.ReadKey().KeyChar;
198                     var aux = Convert.ToInt32(stringConsole) - 48;
199                     if (Project.realAmbient <= 3 && Project.realAmbient >= 0)
200                     {
201                         Project.realAmbient = (ushort)(aux);
202                     }
203                     else
204                     {
205                         Project.realAmbient = 0;
206                         break;
207                     }
208                 }
209
210                 // Stop all the readers and disconnect from them.
211                 foreach (ImpinjReader reader in readers)
212                 {
213                     try
214                     {
215                         reader.Stop();
216                         reader.Disconnect();
217                     }
218                     catch (OctaneSdkException) { }
219                 }
220             }
221             catch (OctaneSdkException e)
```

```
222             {
223                 // Handle Octane SDK errors.
224                 Console.WriteLine("Octane SDK exception: {0}", e.Message);
225             }
226             catch (Exception e)
227             {
228                 // Handle other .NET errors.
229                 Console.WriteLine("Exception : {0}", e.Message);
230             }
231
232
233             // Wait for the user to press enter.
234             Console.WriteLine("Press enter to exit.");
235             Console.ReadKey();
236         }
237
238
239
240         private static void Captura_tags(ImpinjReader sender, TagReport
              report)
241         {
242             foreach (Tag tag in report)
243             {
244                 if (Project.IsTagRegistered(tag) && tag.PeakRssiInDbm >
                      GlobalData.RSSILowPassFilter && Math.Abs
                      (tag.RfDopplerFrequency) > 0.5 )
245                 {
246                     Project.ReadingCardholderTag(tag, sender.Name);
247                     Project.ProcessCardholderData(tag, sender.Name);
248                     var individuo = Project.GetCardholder(tag.Epc.ToString());
249
250                     // Writes data to file
251                     GlobalData.filehandler.WriteToFile(individuo,
                        tag.Epc.ToString(), sender.Name, tag.AntennaPortNumber);
252
253                     // Debug
254                     // Console.WriteLine("RSSI: {0}, Doppler: {1}, 0:{2}, 2:
                        {3}, 3:{4}, 4:{5}, 5:{6}, 6:{7}", tag.PeakRssiInDbm,
                        tag.RfDopplerFrequency, individuo.GetAmbient(0).GetName(),
                         individuo.GetAmbient(2).GetName(), individuo.GetAmbient
                        (3).GetName(), individuo.GetAmbient(4).GetName(),
                        individuo.GetAmbient(5).GetName(), individuo.GetAmbient
                        (6).GetName());
255                     // Debug.end
256                     Console.WriteLine("Name: {0},    Peaks:{1},     Dopppler:
                        {2},    Combined:{3}", individuo.GetName(),
                        individuo.GetAmbient(0).GetName(), individuo.GetAmbient
                        (5).GetName(), individuo.GetAmbient(6).GetName());
257
258                 }
259             }
260         }
261
262
263
264
```

```
265
266         }
267     }
```

```
1  using System;
2  using System.Collections;
3  using System.Collections.Generic;
4  using Impinj.OctaneSdk;
5
6
7  namespace TG2_RFID
8  {
9      public class Project
10     {
11         public static ushort realAmbient = 0;
12
13         /// <summary>
14         /// The map of registered people in the project.
15         /// </summary>
16         protected static volatile Dictionary<string, Cardholder>
             registeredPeople = new Dictionary<string, Cardholder>();
17
18         /// <summary>
19         /// The map of registed ambients in the project.
20         /// </summary>
21         protected static volatile Dictionary<ushort, Ambient> registerAmbient
             = new Dictionary<ushort, Ambient>();
22
23         /// <summary>
24         /// The map of registed transitions in the project.
25         /// </summary>
26         protected static volatile Dictionary<Tuple<string, ushort>,
             Transition> registerTransition = new Dictionary<Tuple<string,
             ushort>, Transition>();
27
28
29         /// <summary>
30         /// Registers a new cardholder given an tag epc.
31         /// </summary>
32         /// <param name="valueEPC">tag EPC as an string epc.</param>
33         /// <param name="person">Person.</param>
34         public static void RegisterNewCardholder(string valueEPC, Cardholder
             person)
35         {
36             person.SetCardholderEPC(valueEPC);
37             registeredPeople.Add(valueEPC, person);
38         }
39
40         /// <summary>
41         /// Registers a new ambient given an antenna.
42         /// </summary>
43         /// <param name="antenna">Antenna.</param>
44         /// <param name="ambient">Ambient.</param>
45         public static void RegisterNewAmbient(ushort roomNumber, Ambient
             ambient)
46         {
47             registerAmbient.Add(roomNumber, ambient);
48         }
49
50         /// <summary>
```

```
51          /// Registers a new ambient given an antenna.
52          /// </summary>
53          /// <param name="antenna">Antenna.</param>
54          /// <param name="transition">Ambient.</param>
55          public static void RegisterNewTransition(Tuple<String, ushort>
              antenna, Transition transition)
56          {
57              registerTransition.Add(antenna, transition);
58          }
59
60          public static void RegisterNewCardholder(string EPC, string name)
61          {
62              var cardholder = new Cardholder(name);
63              Project.registeredPeople.Add(EPC, cardholder);
64              foreach (var transition in Project.registerTransition.Values)
65              {
66                  var ant1 = transition.GetAtributes1stAmb().Item2;
67                  var ant2 = transition.GetAtributes2ndAmb().Item2;
68                  if (!cardholder.curvesPowerReadingsDictionary.ContainsKey
                      (ant1))
69                  {
70                      cardholder.curvesPowerReadingsDictionary.Add(ant1, new
                        Curve());
71                  }
72                  if (!cardholder.curvesPowerReadingsDictionary.ContainsKey
                      (ant2))
73                  {
74                      cardholder.curvesPowerReadingsDictionary.Add(ant2, new
                        Curve());
75                  }
76                  if (!
                      cardholder.curvesDoplerFrequencyReadingsDictionary.ContainsK
                      ey(ant1))
77                  {
78                      cardholder.curvesDoplerFrequencyReadingsDictionary.Add
                        (ant1, new Curve());
79                  }
80                  if (!
                      cardholder.curvesDoplerFrequencyReadingsDictionary.ContainsK
                      ey(ant2))
81                  {
82                      cardholder.curvesDoplerFrequencyReadingsDictionary.Add
                        (ant2, new Curve());
83                  }
84              }
85
86          }
87
88          public static void PopulateProjectCardholders()
89          {
90              Project.registeredPeople.Clear();
91              Project.RegisterNewCardholder("E200 001B 2609 0147 0780 7C25",
                  "Maria Beatriz");
92              Project.RegisterNewCardholder("E200 001B 2609 0147 0510 7BBD",
                  "Andre");
93              Project.RegisterNewCardholder("E200 001B 2609 0147 0460 7BA5",
```

```
                "Jese");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0450 7B99",
                "Marquemi");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0380 7B85",
                "Caixeta");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0910 7C5D",
                "Geordana");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0850 7C39",
                "Takashi");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0840 7C31",
                "Luan");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0710 7C0D",
                "Redy");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0660 7BF5",
                "Mayara");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0900 7C55",
                "Isa");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0390 7B8D",
                "Hooper");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0720 7C01",
                "Joao");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0600 7BD1",
                "Anastacia");
            Project.RegisterNewCardholder("E200 001B 2609 0147 1100 7CA5",
                "Lucas");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0650 7BE9",
                "Renato");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0790 7C2D",
                "Aline");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0590 7BDD",
                "Artur");
            Project.RegisterNewCardholder("E200 001B 2609 0147 1040 7C81",
                "Marina");
            Project.RegisterNewCardholder("E200 001B 2609 0147 0520 7BB1",
                "Jose");
            //GlobalDataReader1.Cadastro.Add("AD08 3003 4604 3152 2C00 0086",
                "Tag exemplo impinj");
        }

        /// <summary>
        /// Checks whether the seen tag is registered.
        /// </summary>
        /// <returns><c>true</c>, if tag registered was ised, <c>false</c>
          otherwise.</returns>
        /// <param name="tag">Tag.</param>
        public static bool IsTagRegistered(Tag tag)
        {
            return registeredPeople.ContainsKey(tag.Epc.ToString()); ;
        }

        /// <summary>
        /// Gets cardholder from tag
        /// </summary>
        public static void ReadingCardholderTag(Tag tag, String senderName)
        {
            registeredPeople.TryGetValue(tag.Epc.ToString(), out Cardholder
```

```
                cardholder);
130
131             cardholder.ReadingCardholderTag(tag, senderName);
132         }
133
134         public static void ProcessDataGiveTransition(Transition transition,
              Tuple<string, ushort> antennaPersonAt, Cardholder person, Tag tag,
              string senderName)
135         {
136             var otherAntenna = transition.GetOtherAntenna(antennaPersonAt);
137             var powerCurveLastAntenna = person.GetPowerCurve(antennaPersonAt);
138             var powerCurveOtherAntenna = person.GetPowerCurve(otherAntenna);
139             var dopplerCurveLastAntenna = person.GetDopplerEffectCurve
                  (antennaPersonAt);
140             var dopplerCurveOtherAntenna = person.GetDopplerEffectCurve
                  (otherAntenna);
141
142             /*
143              * Compare Peaks Time
144              */
145
146             /*
147              * Compare Peaks Time and value
148              */
149
150             /*
151              * Compare last value RSSI
152              */
153
154             /*
155              * Compare mean / median
156              */
157
158             /*
159              * Compare Doppler transition point
160              */
161
162             // Compare powerCurve peaks
163             var peakListLast = powerCurveLastAntenna.CalculatePeaks();
164             var peakListOther = powerCurveOtherAntenna.CalculatePeaks();
165             var maxLastAntenna = powerCurveLastAntenna.GetCurveMaxPoint();
166             var maxOtherAntenna = powerCurveOtherAntenna.GetCurveMaxPoint();
167
168
169             /*
170              * Compare RSSI Peaks Time
171              */
172             if (powerCurveLastAntenna.CompareCurveLastPeak
                  (powerCurveOtherAntenna))
173             {
174                 // sets ambient to cardholder
175                 registeredPeople.TryGetValue(tag.Epc.ToString(), out
                      Cardholder cardholder);
176                 cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                      (antennaPersonAt), 0);
177             }
```

```
178             else
179             {
180                 // sets ambient to cardholder
181                 registeredPeople.TryGetValue(tag.Epc.ToString(), out
                      Cardholder cardholder);
182                 cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                      (otherAntenna), 0);
183             }
184
185             /*
186              * Compare Peaks Time and value
187              */
188             //if ((peakListLast.Count > 0 && peakListOther.Count > 0 &&
                  (peakListLast[peakListLast.Count - 1].Item2 > peakListOther
                  [peakListOther.Count - 1].Item2))
189             //    || (peakListLast.Count == 0 || peakListOther.Count == 0 &&
                  maxLastAntenna.Item1 > maxOtherAntenna.Item1 &&
                  maxLastAntenna.Item2 > maxOtherAntenna.Item2))
190             //{
191             //    // sets ambient to cardholder
192             //    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                  Cardholder cardholder);
193             //    cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                  (antennaPersonAt),1);
194             //}
195             //else
196             //{
197             //    // sets ambient to cardholder
198             //    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                  Cardholder cardholder);
199             //    cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                  (otherAntenna),1);
200             //}
201
202             /*
203              * Compare last value RSSI
204              */
205             if (powerCurveLastAntenna.CompareCurveLastPeak
                  (powerCurveOtherAntenna))
206             {
207                 //sets ambient to cardholder
208                 registeredPeople.TryGetValue(tag.Epc.ToString(), out
                      Cardholder cardholder);
209                 cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                      (antennaPersonAt), 2);
210             }
211             else
212             {
213                 //sets ambient to cardholder
214                 registeredPeople.TryGetValue(tag.Epc.ToString(), out
                      Cardholder cardholder);
215                 cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                      (otherAntenna), 2);
216             }
217
218             /*
```

```
219                 * Compare mean
220                 */
221                if (powerCurveLastAntenna.CompareCurveMeans
                     (powerCurveOtherAntenna))
222                {
223                    //sets ambient to cardholder
224                    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                         Cardholder cardholder);
225                    cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                         (antennaPersonAt), 3);
226                }
227                else
228                {
229                    //sets ambient to cardholder
230                    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                         Cardholder cardholder);
231                    cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                         (otherAntenna), 3);
232                }
233
234                /*
235                 * Compare meadian
236                 */
237                if (powerCurveLastAntenna.CompareCurveMedians
                     (powerCurveOtherAntenna))
238                {
239                    //sets ambient to cardholder
240                    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                         Cardholder cardholder);
241                    cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                         (antennaPersonAt), 4);
242                }
243                else
244                {
245                    //sets ambient to cardholder
246                    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                         Cardholder cardholder);
247                    cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                         (otherAntenna), 4);
248                }
249
250                /*
251                 * Compare Doppler transition point
252                 */
253                if (!Double.IsNaN(dopplerCurveLastAntenna.CalculateCrossingPoint
                     ().Item1) &&
254                    !Double.IsNaN(dopplerCurveOtherAntenna.CalculateCrossingPoint
                         ().Item1) &&
255                    dopplerCurveLastAntenna.CalculateCrossingPoint().Item1 >
                         dopplerCurveOtherAntenna.CalculateCrossingPoint().Item1)
256                {
257                    //sets ambient to cardholder
258                    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                         Cardholder cardholder);
259                    cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                         (antennaPersonAt), 5);
```

```
260             }
261             else if (!Double.IsNaN
                  (dopplerCurveLastAntenna.CalculateCrossingPoint().Item1) &&
262               !Double.IsNaN(dopplerCurveOtherAntenna.CalculateCrossingPoint
                    ().Item1) &&
263               dopplerCurveLastAntenna.CalculateCrossingPoint().Item1 <
                    dopplerCurveOtherAntenna.CalculateCrossingPoint().Item1)
264             {
265                 //sets ambient to cardholder
266                 registeredPeople.TryGetValue(tag.Epc.ToString(), out
                      Cardholder cardholder);
267                 cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                      (otherAntenna), 5);
268             }
269             else
270             {
271                 //if ((peakListLast.Count > 0 && peakListOther.Count > 0 &&
                      (peakListLast[peakListLast.Count - 1].Item1 > peakListOther
                      [peakListOther.Count - 1].Item1))
272                 //|| (peakListLast.Count == 0 || peakListOther.Count == 0 &&
                      maxLastAntenna.Item1 > maxOtherAntenna.Item1 &&
                      maxLastAntenna.Item2 > maxOtherAntenna.Item2))
273                 //{
274                 //    // sets ambient to cardholder
275                 //    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                        Cardholder cardholder);
276                 //    cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                        (antennaPersonAt), 5);
277                 //}
278                 //else
279                 //{
280                 //    // sets ambient to cardholder
281                 //    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                        Cardholder cardholder);
282                 //    cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                        (otherAntenna), 5);
283                 //}
284             }
285
286
287             /*
288              * Compare Doppler transition point and RSSI peaks
289              */
290
291             if ((peakListLast.Count > 0 && peakListOther.Count > 0 &&
                  (peakListLast[peakListLast.Count - 1].Item1 > peakListOther
                  [peakListOther.Count - 1].Item1) &&
292                 (!Double.IsNaN(dopplerCurveLastAntenna.CalculateCrossingPoint
                      ().Item1) &&
293                 !Double.IsNaN(dopplerCurveOtherAntenna.CalculateCrossingPoint
                      ().Item1) &&
294                 dopplerCurveLastAntenna.CalculateCrossingPoint().Item1 >
                      dopplerCurveOtherAntenna.CalculateCrossingPoint().Item1)))
295             //|| (peakListLast.Count == 0 || peakListOther.Count == 0 &&
                  maxLastAntenna.Item1 > maxOtherAntenna.Item1 &&
                  maxLastAntenna.Item2 > maxOtherAntenna.Item2))
```

```
296             {
297                 // sets ambient to cardholder
298                 registeredPeople.TryGetValue(tag.Epc.ToString(), out
                      Cardholder cardholder);
299                 cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                      (antennaPersonAt), 6);
300             }
301             else if ((peakListLast.Count > 0 && peakListOther.Count > 0 &&
                  (peakListLast[peakListLast.Count - 1].Item1 <= peakListOther
                  [peakListOther.Count - 1].Item1) &&
302                 (!Double.IsNaN(dopplerCurveLastAntenna.CalculateCrossingPoint
                      ().Item1) &&
303                 !Double.IsNaN(dopplerCurveOtherAntenna.CalculateCrossingPoint
                      ().Item1) &&
304                 dopplerCurveLastAntenna.CalculateCrossingPoint().Item1 <=
                      dopplerCurveOtherAntenna.CalculateCrossingPoint().Item1)))
305             {
306                 // sets ambient to cardholder
307                 registeredPeople.TryGetValue(tag.Epc.ToString(), out
                      Cardholder cardholder);
308                 cardholder.SetAmbient(transition.GetAmb4GivenAntenna
                      (otherAntenna), 6);
309             }
310
311
312
313
314
315
316
317             //if (powerCurveLastAntenna.GetSize() > 4 ||
318             //    powerCurveLastAntenna.GetSize() > 4)
319             //{
320             //    int TESTANDO = 0;
321             //}
322
323             //if ((peakListLast.Count > 0 && peakListOther.Count > 0 &&
                  (peakListLast[peakListLast.Count - 1].Item1 > peakListOther
                  [peakListOther.Count - 1].Item1))
324             //    || (peakListLast.Count == 0 || peakListOther.Count == 0 &&
                  maxLastAntenna.Item1 > maxOtherAntenna.Item1))
325             //if (maxLastAntenna.Item1 > maxOtherAntenna.Item1)
326             //if (powerCurveLastAntenna.CalculateMeanY() >
                  powerCurveOtherAntenna.CalculateMeanY())
327             //if (powerCurveLastAntenna.GetCurveLastValue() >
                  powerCurveOtherAntenna.GetCurveLastValue())
328             //{
329             //    sets ambient to cardholder
330             //    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                  Cardholder cardholder);
331             //    cardholder.SetCurrAmbient(transition.GetAmb4GivenAntenna
                  (antennaPersonAt));
332             //}
333             //else
334             //{
335             //    sets ambient to cardholder
```

```
336             //    registeredPeople.TryGetValue(tag.Epc.ToString(), out
                  Cardholder cardholder);
337             //    cardholder.SetCurrAmbient(transition.GetAmb4GivenAntenna
                  (otherAntenna));
338             //}
339
340             person.SetCurrAmbient(Project.GetAmbientInstance
                  (Project.realAmbient));
341         }
342
343         // TODO
344         // Aqui vamos processar a curva já populada!
345         // Processa o cardholder data
346         public static void ProcessCardholderData(Tag tag, string senderName)
347         {
348             //get curves
349             registeredPeople.TryGetValue(tag.Epc.ToString(), out Cardholder
                  person);
350             var antennaPersonAt = Tuple.Create<string, ushort>(senderName,
                  tag.AntennaPortNumber);
351             var transition = Project.GetTransitionInstance(antennaPersonAt);
352
353             var ambient = transition.GetAmb4GivenAntenna(antennaPersonAt);
354
355             Tuple.Create<string, ushort>("Reader #1", 2);
356             Tuple.Create<string, ushort>("Reader #2", 1);
357             Tuple.Create<string, ushort>("Reader #2", 2);
358             Tuple.Create<string, ushort>("Reader #3", 1);
359             Tuple.Create<string, ushort>("Reader #3", 2);
360
361             if (ambient.GetName() == ("Area_Externa(0)") &&
362                 transition != Project.GetTransitionInstance
                      (Tuple.Create<string, ushort>("Reader #1", 1)))
363             {
364                 return;
365             }
366             else if (ambient.GetName() == ("Sala_Reuniao(2)") &&
367                 transition != Project.GetTransitionInstance
                      (Tuple.Create<string, ushort>("Reader #2", 1)))
368             {
369                 return;
370             }
371             else if (ambient.GetName() == ("Corredor_Baias(3)") &&
372                 transition != Project.GetTransitionInstance
                      (Tuple.Create<string, ushort>("Reader #3", 2)))
373             {
374                 return;
375             }
376             //else if (ambient.GetName() == ("Sala_Principal(1)")
377
378
379
380             ProcessDataGiveTransition(transition, antennaPersonAt, person,
                  tag, senderName);
381         }
382
```

```
383         /// <summary>
384         /// Returns Ambient type instance from dictionary according to key
              given
385         /// </summary>
386         public static Ambient GetAmbientInstance (ushort ambientKey)
387         {
388             registerAmbient.TryGetValue(ambientKey, out Ambient
                ambientInstance);
389             return ambientInstance;
390         }
391
392         /// <summary>
393         /// Returns Transition type instance from dictionary according to key
              given
394         /// </summary>
395         public static Transition GetTransitionInstance(Tuple<string, ushort>
              antennaID)
396         {
397             registerTransition.TryGetValue(antennaID, out Transition
                transitionInstance);
398             return transitionInstance;
399         }
400
401         /// <summary>
402         /// Getter people
403         /// </summary>
404         ///
405         public static Cardholder GetCardholder(string tag)
406         {
407             registeredPeople.TryGetValue(tag, out Cardholder cardholderObj);
408             return cardholderObj;
409         }
410
411
412
413
414
415
416     }
417 }
418
```

```
1  using System;
2  using System.Collections.Generic;
3  using System.Text;
4
5  namespace TG2_RFID
6  {
7      public class Transition
8      {
9          /// <summary>
10         /// Holds the ambients the transition connects
11         /// </summary>
12         protected Ambient ambient1, ambient2;
13
14         /// <summary>
15         /// Holds the antennas that make the transition
16         /// </summary>
17         protected Tuple<string, ushort> antenna1, antenna2;
18
19         /// <summary>
20         /// Setter for the Ambients of the transition.
21         /// </summary>
22         public void SetAmbients2Transition(Ambient amb1, Ambient amb2)
23         {
24             ambient1 = amb1;
25             ambient2 = amb2;
26         }
27
28         /// <summary>
29         /// Setter for the Antennas of the transition.
30         /// </summary>
31         public void SetAntennas2Transition(string firstReader, ushort
             firstAntenna, string secndReader, ushort secndAntenna)
32         {
33             Tuple.Create<string, ushort>(firstReader, firstAntenna);
34             Tuple.Create<string, ushort>(secndReader, secndAntenna);
35         }
36
37         /// <summary>
38         /// Getter for first Ambient and Antenna
39         /// </summary>
40         public Tuple<Ambient, Tuple<string, ushort>> GetAtributes1stAmb()
41         {
42             return Tuple.Create<Ambient, Tuple<string, ushort>>(ambient1,
                 antenna1);
43         }
44
45         /// <summary>
46         /// Getter for second Ambient and Antenna
47         /// </summary>
48         public Tuple<Ambient, Tuple<string, ushort>> GetAtributes2ndAmb()
49         {
50             return Tuple.Create<Ambient, Tuple<string, ushort>>(ambient2,
                 antenna2);
51         }
52
53         /// <summary>
```

```
54          /// Setter for transition
55          /// </summary>
56          public Transition (Ambient amb1, string reader1, ushort ant1, Ambient
              amb2, string reader2, ushort ant2)
57          {
58              antenna1 = Tuple.Create<string, ushort>(reader1, ant1);
59              antenna2 = Tuple.Create<string, ushort>(reader2, ant2);
60              ambient1 = amb1;
61              ambient2 = amb2;
62          }
63
64          /// <summary>
65          /// Gets the second antenna, given the first one
66          /// </summary>
67          public Tuple<string, ushort> GetOtherAntenna(Tuple <string, ushort>
              givenAntenna)
68          {
69              if (antenna1.Item1 == givenAntenna.Item1 && antenna1.Item2 ==
                  givenAntenna.Item2)
70              //if (antenna1.Equals(givenAntenna))
71              {
72                  return antenna2;
73              }
74              else
75              {
76                  return antenna1;
77              }
78          }
79
80
81          /// <summary>
82          /// Getter for Ambient related to antenna
83          /// </summary>
84          public Ambient GetAmb4GivenAntenna(Tuple <string, ushort> ant)
85          {
86              if (ant.Item1 == antenna1.Item1 && ant.Item2 == antenna1.Item2)
87              {
88                  return ambient1;
89              } else if (ant.Item1 == antenna2.Item1 && ant.Item2 ==
                    antenna2.Item2)
90              {
91                  return ambient2;
92              } else
93              {
94                  throw new Exception ();
95              }
96          }
97
98      }
99  }
100
```

III. REGISTRO DE DADOS GERADOS NO TESTE

As páginas apresentadas neste anexo apresentam o registro dos dados gerados no teste executado. Durante o teste, uma pessoa portando uma TAG percorre a sala do LARA, caminhando em velocidade constante entre os ambientes.

Nome	EPC	Leitora	Antena	Data	Horario	Ambiente Correto	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:32:42:7065	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:32:42:7106	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:42:7138	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:42:8942	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:32:42:8984	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:32:43:3937	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:43:6434	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:43:6449	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:43:8964	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:44:1461	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:32:47:5954	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:52:8937	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:52:8960	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:53:3983	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:32:53:6442	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:32:53:6456	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:56:1432	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:57:1492	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:57:6450	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:32:58:6443	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:32:58:8956	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:32:59:3939	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:59:6440	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:32:59:6466	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:32:59:6483	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:33:00:1446	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:33:00:3956	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:33:00:6434	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:33:00:6449	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:33:00:8944	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:33:01:1436	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:33:01:3941	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:10:0961	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:10:0974	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:10:3475	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:10:3487	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:10:5982	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:10:6031	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:16:5963	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:16:5977	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:16:8460	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:17:0969	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:18:5961	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:19:0978	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:19:8464	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:20:0961	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:21:3477	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:23:3477	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:25:5965	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:27:0964	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:29:8466	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:30:0965	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:30:0988	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:30:5970	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:36:5972	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:36:6016	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:36:6117	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:36:8475	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:36:8514	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:37:0976	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:37:3470	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:37:3492	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:37:3593	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:37:3627	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:37:5966	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:37:5980	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:37:5998	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:37:6051	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:37:8467	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:37:8477	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:38:0966	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:38:0977	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:41:8470	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:42:0968	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:42:3475	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:42:3502	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:42:3565	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:42:5966	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:42:8483	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:42:8646	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:42:8788	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:42:8938	Corredor_Baias(3)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:43:0974	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:43:1062	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:43:1096	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:43:1125	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:43:3472	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	2	10/12/2019	02:33:43:3486	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:43:5968	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:43:5979	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:43:6029	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:43:6041	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:49:5180	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:49:7685	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:49:7709	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:50:0184	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:50:0213	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:50:2692	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:50:2761	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:50:5188	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:50:5298	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:50:5348	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:50:7678	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:51:0190	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:51:0239	Sala_Reuniao(2)	

Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:51:2706	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:51:2765	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:51:2867	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:52:0191	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:52:0218	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:55:7689	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:55:7812	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:56:2687	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:56:2716	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:56:5184	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:56:5195	Sala_Reuniao(2)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:56:7686	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	1	10/12/2019	02:33:57:0199	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:57:2716	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:57:2806	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:57:5194	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:58:0201	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #2	2	10/12/2019	02:33:58:0251	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:58:3490	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #3	1	10/12/2019	02:33:58:5972	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:34:05:6446	Sala_Principal(1)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:34:05:8943	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	2	10/12/2019	02:34:06:1438	Area_Externa(0)	
Jose	E200 001B 2609 0147 0520 7BB1	Reader #1	1	10/12/2019	02:34:06:3937	Area_Externa(0)	

	Curva RSSI										
	-58,5	0	0	0	0	0	0	0	0	0	0
	-58,5	-57	0	0	0	0	0	0	0	0	0
	-58	0	0	0	0	0	0	0	0	0	0
	-58	-57,5	0	0	0	0	0	0	0	0	0
	-58,5	-57	-59,5	0	0	0	0	0	0	0	0
	-58,5	-57	-59,5	-57,83333333	0	0	0	0	0	0	0
	-58	-57,5	-58,5	0	0	0	0	0	0	0	0
	-58	-57,5	-58,5	-56,66666667	0	0	0	0	0	0	0
	-58	-57,5	-58,5	-56,66666667	-56,38888889	0	0	0	0	0	0
	-58	-57,5	-58,5	-56,66666667	-56,38888889	-57,18518519	0	0	0	0	0
	-58	0	0	0	0	0	0	0	0	0	0
	-58	-57,5	-58,5	-56,66666667	-56,38888889	-57,18518519	-56,85802469	0	0	0	0
	-58	-57,5	-58,5	-56,66666667	-56,38888889	-57,18518519	-56,85802469	-56,34773663	0	0	0
	-58	-57,5	-58,5	-56,66666667	-56,38888889	-57,18518519	-56,85802469	-56,34773663	-56,06858711	0	0
	-58,5	-57	-59,5	-57,83333333	-58,61111111	0	0	0	0	0	0
	-58,5	-57	-59,5	-57,83333333	-58,61111111	-58,31481481	0	0	0	0	0
	-58	-57,5	-58,5	-56,66666667	-56,38888889	-57,18518519	-56,85802469	-56,34773663	-56,06858711	-57,13877458	0
	-58	-57,5	-58,5	-56,66666667	-56,38888889	-57,18518519	-56,85802469	-56,34773663	-56,06858711	-57,13877458	-57,40245389
	-57,5	-58,5	-56,66666667	-56,38888889	-57,18518519	-56,85802469	-56,34773663	-56,06858711	-57,13877458	-57,40245389	-57,01374282
	-58,5	-57	-59,5	-57,83333333	-58,61111111	-58,31481481	-58,47530864	0	0	0	0
	-58,5	-57	-59,5	-57,83333333	-58,61111111	-58,31481481	-58,47530864	-58,26337449	0	0	0
	-58,5	-57	-59,5	-57,83333333	-58,61111111	-58,31481481	-58,47530864	-58,26337449	-58,74622771	0	0
	-58,5	-56,66666667	-56,38888889	-57,18518519	-56,85802469	-56,34773663	-56,06858711	-57,13877458	-57,40245389	-57,01374282	-57,80539891
	-56,66666667	-56,38888889	-57,18518519	-56,85802469	-56,34773663	-56,06858711	-57,13877458	-57,40245389	-57,01374282	-57,80539891	-57,93971391
	-58,5	-57	-59,5	-57,83333333	-58,61111111	-58,31481481	-58,47530864	-58,26337449	-58,74622771	-58,6698674	0
	-56,38888889	-57,18518519	-56,85802469	-56,34773663	-56,06858711	-57,13877458	-57,40245389	-57,01374282	-57,80539891	-57,93971391	-58,41503761
	-58,5	-57	-59,5	-57,83333333	-58,61111111	-58,31481481	-58,47530864	-58,26337449	-58,74622771	-58,6698674	-58,9720317
	-57,18518519	-56,85802469	-56,34773663	-56,06858711	-57,13877458	-57,40245389	-57,01374282	-57,80539891	-57,93971391	-58,41503761	-57,95158384
	-56,85802469	-56,34773663	-56,06858711	-57,13877458	-57,40245389	-57,01374282	-57,80539891	-57,93971391	-58,41503761	-57,95158384	-58,12220715
	-56,34773663	-56,06858711	-57,13877458	-57,40245389	-57,01374282	-57,80539891	-57,93971391	-58,41503761	-57,95158384	-58,12220715	-57,19126366
	-56,06858711	-57,13877458	-57,40245389	-57,01374282	-57,80539891	-57,93971391	-58,41503761	-57,95158384	-58,12220715	-57,19126366	-58,10449027
	-57,13877458	-57,40245389	-57,01374282	-57,80539891	-57,93971391	-58,41503761	-57,95158384	-58,12220715	-57,19126366	-58,10449027	-57,93191798
	-58	-57	0	0	0	0	0	0	0	0	0
	-58	-57	-58	0	0	0	0	0	0	0	0
	-58	-57	-58	-57,66666667	0	0	0	0	0	0	0
	-59	0	0	0	0	0	0	0	0	0	0
	-58	-57	-58	-57,66666667	-57,55555556	0	0	0	0	0	0
	-58	-57	-58	-57,66666667	-57,55555556	-57,40740741	0	0	0	0	0
	-59	-59,5	0	0	0	0	0	0	0	0	0
	-59	-59,5	-59,5	0	0	0	0	0	0	0	0
	-59	-59,5	-59,5	-59,33333333	0	0	0	0	0	0	0
	-59	-59,5	-59,5	-59,33333333	-59,11111111	0	0	0	0	0	0
	-59	-59,5	-59,5	-59,33333333	-59,11111111	-58,64814815	0	0	0	0	0
	-59	-59,5	-59,5	-59,33333333	-59,11111111	-58,64814815	-58,58641975	0	0	0	0
	-59	-59,5	-59,5	-59,33333333	-59,11111111	-58,64814815	-58,58641975	-58,41152263	0	0	0
	-59	-59,5	-59,5	-59,33333333	-59,11111111	-58,64814815	-58,58641975	-58,41152263	-58,6659808	0	0
	-59	-59,5	-59,5	-59,33333333	-59,11111111	-58,64814815	-58,58641975	-58,41152263	-58,6659808	-58,69250114	0
	-59	-59,5	-59,5	-59,33333333	-59,11111111	-58,64814815	-58,58641975	-58,41152263	-58,6659808	-58,69250114	-58,78616065
	-59,5	-59,5	-59,33333333	-59,11111111	-58,64814815	-58,58641975	-58,41152263	-58,6659808	-58,69250114	-58,78616065	-58,3262206
	-59,5	-59,33333333	-59,11111111	-58,64814815	-58,58641975	-58,41152263	-58,6659808	-58,69250114	-58,78616065	-58,3262206	-58,37079375
	-59,33333333	-59,11111111	-58,64814815	-58,58641975	-58,41152263	-58,6659808	-58,69250114	-58,78616065	-58,3262206	-58,37079375	-58,23233811
	-59,11111111	-58,64814815	-58,58641975	-58,41152263	-58,6659808	-58,69250114	-58,78616065	-58,3262206	-58,37079375	-58,23233811	-58,20104395
	-58,64814815	-58,58641975	-58,41152263	-58,6659808	-58,69250114	-58,78616065	-58,3262206	-58,37079375	-58,23233811	-58,20104395	-57,14446069
	-58	-57	-58	-57,66666667	-57,55555556	-57,40740741	-57,32098765	0	0	0	0
	-58	-57	-58	-57,66666667	-57,55555556	-57,40740741	-57,32098765	-56,57613169	0	0	0
	-58	-57	-58	-57,66666667	-57,55555556	-57,40740741	-57,32098765	-56,57613169	-56,13237311	0	0
	-58	-57	-58	-57,66666667	-57,55555556	-57,40740741	-57,32098765	-56,57613169	-56,13237311	-54,73616827	0
	-58	-57	-58	-57,66666667	-57,55555556	-57,40740741	-57,32098765	-56,57613169	-56,13237311	-54,73616827	-53,95618046
	-57	-58	-57,66666667	-57,55555556	-57,40740741	-57,32098765	-56,57613169	-56,13237311	-54,73616827	-53,95618046	-54,06411624
	-58	-57,66666667	-57,55555556	-57,40740741	-57,32098765	-56,57613169	-56,13237311	-54,73616827	-53,95618046	-54,06411624	-54,00676557
	-57,66666667	-57,55555556	-57,40740741	-57,32098765	-56,57613169	-56,13237311	-54,73616827	-53,95618046	-54,06411624	-54,00676557	-54,52362727
	-58,58641975	-58,41152263	-58,6659808	-58,69250114	-58,78616065	-58,3262206	-58,37079375	-58,23233811	-58,20104395	-57,14446069	-56,94850155
	-58,41152263	-58,6659808	-58,69250114	-58,78616065	-58,3262206	-58,37079375	-58,23233811	-58,20104395	-57,14446069	-56,94850155	-57,53098741
	-57,55555556	-57,40740741	-57,32098765	-56,57613169	-56,13237311	-54,73616827	-53,95618046	-54,06411624	-54,00676557	-54,52362727	-53,67679761
	-57,40740741	-57,32098765	-56,57613169	-56,13237311	-54,73616827	-53,95618046	-54,06411624	-54,00676557	-54,52362727	-53,67679761	-54,06680829
	-57,32098765	-56,57613169	-56,13237311	-54,73616827	-53,95618046	-54,06411624	-54,00676557	-54,52362727	-53,67679761	-54,06680829	-53,4145353
	-56,57613169	-56,13237311	-54,73616827	-53,95618046	-54,06411624	-54,00676557	-54,52362727	-53,67679761	-54,06680829	-53,4145353	-52,82711453
	-58,6659808	-58,69250114	-58,78616065	-58,3262206	-58,37079375	-58,23233811	-58,20104395	-57,14446069	-56,94850155	-57,53098741	-57,15982965
	-58,69250114	-58,78616065	-58,3262206	-58,37079375	-58,23233811	-58,20104395	-57,14446069	-56,94850155	-57,53098741	-57,15982965	-57,23027236
	-56,13237311	-54,73616827	-53,95618046	-54,06411624	-54,00676557	-54,52362727	-53,67679761	-54,06680829	-53,4145353	-52,82711453	-54,91388328
	-58,78616065	-58,3262206	-58,37079375	-58,23233811	-58,20104395	-57,14446069	-56,94850155	-57,53098741	-57,15982965	-57,23027236	-56,79670067
	-58,3262206	-58,37079375	-58,23233811	-58,20104395	-57,14446069	-56,94850155	-57,53098741	-57,15982965	-57,23027236	-56,79670067	-57,34232434
	-58,37079375	-58,23233811	-58,20104395	-57,14446069	-56,94850155	-57,53098741	-57,15982965	-57,23027236	-56,79670067	-57,34232434	-57,879675
	-58,23233811	-58,20104395	-57,14446069	-56,94850155	-57,53098741	-57,15982965	-57,23027236	-56,79670067	-57,34232434	-57,879675	-57,90733312
	-58,20104395	-57,14446069	-56,94850155	-57,53098741	-57,15982965	-57,23027236	-56,79670067	-57,34232434	-57,879675	-57,90733312	-57,59566937
	-57,14446069	-56,94850155	-57,53098741	-57,15982965	-57,23027236	-56,79670067	-57,34232434	-57,879675	-57,90733312	-57,59566937	-57,33433416
	-56,94850155	-57,53098741	-57,15982965	-57,23027236	-56,79670067	-57,34232434	-57,879675	-57,90733312	-57,59566937	-57,33433416	-56,97666785
	-57,53098741	-57,15982965	-57,23027236	-56,79670067	-57,34232434	-57,879675	-57,90733312	-57,59566937	-57,33433416	-56,97666785	-54,270334
	-54,73616827	-53,95618046	-54,06411624	-54,00676557	-54,52362727	-53,67679761	-54,06680829	-53,4145353	-52,82711453	-54,91388328	-55,5803326
	-53,95618046	-54,06411624	-54,00676557	-54,52362727	-53,67679761	-54,06680829	-53,4145353	-52,82711453	-54,91388328	-55,5803326	-56,49807196
	-57,15982965	-57,23027236	-56,79670067	-57,34232434	-57,879675	-57,90733312	-57,59566937	-57,33433416	-56,97666785	-54,270334	-53,91566728
	-57,23027236	-56,79670067	-57,34232434	-57,879675	-57,90733312	-57,59566937	-57,33433416	-56,97666785	-54,270334	-53,91566728	-53,06200043
	-54,06411624	-54,00676557	-54,52362727	-53,67679761	-54,06680829	-53,4145353	-52,82711453	-54,91388328	-55,5803326	-56,49807196	-55,35946819
	-54,00676557	-54,52362727	-53,67679761	-54,06680829	-53,4145353	-52,82711453	-54,91388328	-55,5803326	-56,49807196	-55,35946819	-54,78584672
	-56,79670067	-57,34232434	-57,879675	-57,90733312	-57,59566937	-57,33433416	-56,97666785	-54,270334	-53,91566728	-53,06200043	-54,32588924
	-57,34232434	-57,879675	-57,90733312	-57,59566937	-57,33433416	-56,97666785	-54,270334	-53,91566728	-53,06200043	-54,32588924	-54,29596322
	-54,52362727	-53,67679761	-54,06680829	-53,4145353	-52,82711453	-54,91388328	-55,5803326	-56,49807196	-55,35946819	-54,78584672	-56,21510497
	-57,879675	-57,90733312	-57,59566937	-57,33433416	-56,97666785	-54,270334	-53,91566728	-53,06200043	-54,32588924	-54,29596322	-55,20728415
	-53,67679761	-54,06680829	-53,4145353	-52,82711453	-54,91388328	-55,5803326	-56,49807196	-55,35946819	-54,78584672	-56,21510497	-54,83365056
	-54,06680829	-53,4145353	-52,82711453	-54,91388328	-55,5803326	-56,49807196	-55,35946819	-54,78584672	-56,21510497	-54,83365056	-55,18291851
	-53,4145353	-52,82711453	-54,91388328	-55,5803326	-56,49807196	-55,35946819	-54,78584672	-56,21510497	-54,83365056	-55,18291851	-54,83885636
	-52,82711453	-54,91388328	-55,5803326	-56,49807196	-55,35946819	-54,78584672	-56,21510497	-54,83365056	-55,18291851	-54,83885636	-55,34059162
	-59	0	0	0	0	0	0	0	0	0	0
	-59	-59	0	0	0	0	0	0	0	0	0
	-59	-59	-51,5	0	0	0	0	0	0	0	0
	-59	-59	-51,5	-56,33333333	0	0	0	0	0	0	0
	-59	-59	-51,5	-56,33333333	-54,61111111	0	0	0	0	0	0
	-59	-59	-51,5	-56,33333333	-54,61111111	-54,31481481	0	0	0	0	0
	-59	-59	-51,5	-56,33333333	-54,61111111	-54,31481481	-53,64197531	0	0	0	0
	-59	-59	-51,5	-56,33333333	-54,61111111	-54,31481481	-53,64197531	-53,31893004	0	0	0
	-59	-59	-51,5	-56,33333333	-54,61111111	-54,31481481	-53,64197531	-53,31893004	-52,82030178	0	0
	-59	0	0	0	0	0	0	0	0	0	0
	-59	-59	-51,5	-56,33333333	-54,61111111	-54,31481481	-53,64197531	-53,31893004	-52,82030178	-54,87974394	0
	-59	-57,5	0	0	0	0	0	0	0	0	0
	-59	-57,5	-55,5	0	0	0	0	0	0	0	0

	-59	-57,5	-55,5	-56,5	0	0	0	0	0	0	0
	-59	-59	-51,5	-56,33333333	-54,61111111	-54,31481481	-53,64197531	-53,31893004	-52,82030178	-54,87974394	-54,56668191
	-59	-51,5	-56,33333333	-54,61111111	-54,31481481	-53,64197531	-53,31893004	-52,82030178	-54,87974394	-54,56668191	-55,48214195
	-59	-57,5	-55,5	-56,5	-55,66666667	0	0	0	0	0	0
	-59	-57,5	-55,5	-56,5	-55,66666667	-55,55555556	0	0	0	0	0
	-59	-57,5	-55,5	-56,5	-55,66666667	-55,55555556	-55,74074074	0	0	0	0
	-59	-57,5	-55,5	-56,5	-55,66666667	-55,55555556	-55,74074074	-55,59876543	0	0	0
	-59	-57,5	-55,5	-56,5	-55,66666667	-55,55555556	-55,74074074	-55,59876543	-54,44650206	0	0
	-59	-57,5	-55,5	-56,5	-55,66666667	-55,55555556	-55,74074074	-55,59876543	-54,44650206	-54,18175583	0
	-59	-57,5	-55,5	-56,5	-55,66666667	-55,55555556	-55,74074074	-55,59876543	-54,44650206	-54,18175583	-54,87608596
	-57,5	-55,5	-56,5	-55,66666667	-55,55555556	-55,74074074	-55,59876543	-54,44650206	-54,18175583	-54,87608596	-54,68594726
	-55,5	-56,5	-55,66666667	-55,55555556	-55,74074074	-55,59876543	-54,44650206	-54,18175583	-54,87608596	-54,68594726	-54,35401108
	-56,5	-55,66666667	-55,55555556	-55,74074074	-55,59876543	-54,44650206	-54,18175583	-54,87608596	-54,68594726	-54,35401108	-54,51331945
	-51,5	-56,33333333	-54,61111111	-54,31481481	-53,64197531	-53,31893004	-52,82030178	-54,87974394	-54,56668191	-55,48214195	-55,51627462
	-56,33333333	-54,61111111	-54,31481481	-53,64197531	-53,31893004	-52,82030178	-54,87974394	-54,56668191	-55,48214195	-55,51627462	-54,99947219
	-54,61111111	-54,31481481	-53,64197531	-53,31893004	-52,82030178	-54,87974394	-54,56668191	-55,48214195	-55,51627462	-54,99947219	-55,50524894
	-54,31481481	-53,64197531	-53,31893004	-52,82030178	-54,87974394	-54,56668191	-55,48214195	-55,51627462	-54,99947219	-55,50524894	-56,33490704
	-53,64197531	-53,31893004	-52,82030178	-54,87974394	-54,56668191	-55,48214195	-55,51627462	-54,99947219	-55,50524894	-56,33490704	-57,11338533
	-54,91388328	-55,5803326	-56,49807196	-55,35946819	-54,78584672	-56,21510497	-54,83365056	-55,18291851	-54,83885636	-55,34059162	-56,39314933
	-55,5803326	-56,49807196	-55,35946819	-54,78584672	-56,21510497	-54,83365056	-55,18291851	-54,83885636	-55,34059162	-56,39314933	-57,07791365
	-57,40245389	-57,01374282	-57,80539891	-57,93971391	-58,41503761	-57,95158384	-58,12220715	-57,19126366	-58,10449027	-57,93191798	-58,17880275
	-57,01374282	-57,80539891	-57,93971391	-58,41503761	-57,95158384	-58,12220715	-57,19126366	-58,10449027	-57,93191798	-58,17880275	-58,53690691
	-57,80539891	-57,93971391	-58,41503761	-57,95158384	-58,12220715	-57,19126366	-58,10449027	-57,93191798	-58,17880275	-58,53690691	-58,73856989
	-57	-59,5	-57,83333333	-58,61111111	-58,31481481	-58,47530864	-58,26337449	-58,74622771	-58,6698674	-58,9720317	-57,88063303

	Tempo RSSI											
	0	0	0	0	0	0	0	0	0	0	0	
	0	55,4012	0	0	0	0	0	0	0	0	0	
	61,4164	0	0	0	0	0	0	0	0	0	0	
	61,4164	241,9069	0	0	0	0	0	0	0	0	0	
	0	55,4012	244,7878	0	0	0	0	0	0	0	0	
	0	55,4012	244,7878	741,4085	0	0	0	0	0	0	0	
	61,4164	241,9069	991,0493	0	0	0	0	0	0	0	0	
	61,4164	241,9069	991,0493	992,2347	0	0	0	0	0	0	0	
	61,4164	241,9069	991,0493	992,2347	1242,7769	0	0	0	0	0	0	
	61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	0	0	0	0	0	
	4943,0372	0	0	0	0	0	0	0	0	0	0	
	61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	0	0	0	0	
	61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	0	0	0	
	61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	0	0	
	0	55,4012	244,7878	741,4085	10991,6913	0	0	0	0	0	0	
	0	55,4012	244,7878	741,4085	10991,6913	10993,0574	0	0	0	0	0	
	61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	0	
	61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	
	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	
	0	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	0	0	0	0	
	0	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	16242,8737	0	0	0	
	0	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	16242,8737	16741,5382	0	0	
	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	
	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	
	0	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	16242,8737	16741,5382	16995,8566	0	
	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	
	0	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	16242,8737	16741,5382	16995,8566	17742,8505	
	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	
	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	
	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	
	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	18491,2526	
	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	18491,2526	18741,3122	
	4943,0372	27443,6946	0	0	0	0	0	0	0	0	0	
	4943,0372	27443,6946	27444,9933	0	0	0	0	0	0	0	0	
	4943,0372	27443,6946	27444,9933	27694,5975	0	0	0	0	0	0	0	
	27696,3162	0	0	0	0	0	0	0	0	0	0	
	4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	0	0	0	0	0	0	
	4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	0	0	0	0	0	
	27696,3162	33943,9055	0	0	0	0	0	0	0	0	0	
	27696,3162	33943,9055	33945,2577	0	0	0	0	0	0	0	0	
	27696,3162	33943,9055	33945,2577	34193,5997	0	0	0	0	0	0	0	
	27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	0	0	0	0	0	0	
	27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	0	0	0	0	0	
	27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	0	0	0	0	
	27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	0	0	0	
	27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	0	0	
	27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	0	
	27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	
	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	
	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	
	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	
	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	
	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	
	4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	0	0	0	0	
	4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	0	0	0	
	4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	0	0	
	4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	0	
	4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	
	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	
	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	
	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	
	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,4343
	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	
	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	
	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	
	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	
	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	
	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	
	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	
	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	
	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	
	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	
	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	
	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	
	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	
	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	
	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	
	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	
	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	
	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	
	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	60225,177	
	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	60225,177	60241,3146	
	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	
	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	
	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	60225,177	60241,3146	60456,8282	
	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	60225,177	60241,3146	60456,8282	60459,7947	
	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	
	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	60225,177	60241,3146	60456,8282	60459,7947	60696,1124	
	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	
	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	60945,4313	
	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	60945,4313	60950,3092	
	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	60945,4313	60950,3092	60951,6426	
	66865,6655	0	0	0	0	0	0	0	0	0	0	
	66865,6655	67116,0371	0	0	0	0	0	0	0	0	0	
	66865,6655	67116,0371	67118,1527	0	0	0	0	0	0	0	0	
	66865,6655	67116,0371	67118,1527	67365,9737	0	0	0	0	0	0	0	
	66865,6655	67116,0371	67118,1527	67365,9737	67368,596	0	0	0	0	0	0	
	66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	0	0	0	0	0	
	66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	0	0	0	0	
	66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	0	0	0	
	66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	0	0	
	67881,5174	0	0	0	0	0	0	0	0	0	0	
	66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	0	
	67881,5174	68366,1905	0	0	0	0	0	0	0	0	0	
	67881,5174	68366,1905	68370,7653	0	0	0	0	0	0	0	0	

	67881,5174	68366,1905	68370,7653	68615,9201	0	0	0	0	0	0	0	
	66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	
	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	
	67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	0	0	0	0	0	0	
	67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	0	0	0	0	0	
	67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	0	0	0	0	
	67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	0	0	0	
	67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	0	0	
	67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	73619,0375	0	
	67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	73619,0375	73866,0067	
	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	73619,0375	73866,0067	73867,0761	
	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	73619,0375	73866,0067	73867,0761	74116,2457	
	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	73619,0375	73866,0067	73867,0761	74116,2457	74366,9877	
	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	74618,3672	
	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	74618,3672	74627,7978	
	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	74618,3672	74627,7978	74866,3177	
	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	74618,3672	74627,7978	74866,3177	75367,2235	
	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	74618,3672	74627,7978	74866,3177	75367,2235	75372,1055	
	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	60945,4313	60950,3092	60951,6426	75696,2426	
	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	60945,4313	60950,3092	60951,6426	75696,2426	75944,9133	
	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	18491,2526	18741,3122	82992,172	
	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	18491,2526	18741,3122	82992,172	83241,8603	
	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	18491,2526	18741,3122	82992,172	83241,8603	83491,4789	
	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	16242,8737	16741,5382	16995,8566	17742,8505	83741,3075	

Curva Doppler												
2,9375	0	0	0	0	0	0	0	0	0	0		
2,9375	3,3125	0	0	0	0	0	0	0	0	0		
3,375	0	0	0	0	0	0	0	0	0	0		
3,375	3,25	0	0	0	0	0	0	0	0	0		
2,9375	3,3125	1,9375	0	0	0	0	0	0	0	0		
2,9375	3,3125	1,9375	0,5625	0	0	0	0	0	0	0		
3,375	3,25	1,1875	0	0	0	0	0	0	0	0		
3,375	3,25	1,1875	1,5625	0	0	0	0	0	0	0		
3,375	3,25	1,1875	1,5625	-0,625	0	0	0	0	0	0		
3,375	3,25	1,1875	1,5625	-0,625	-0,5625	0	0	0	0	0		
1,875	0	0	0	0	0	0	0	0	0	0		
3,375	3,25	1,1875	1,5625	-0,625	-0,5625	5	0	0	0	0		
3,375	3,25	1,1875	1,5625	-0,625	-0,5625	5	3,6875	0	0	0		
3,375	3,25	1,1875	1,5625	-0,625	-0,5625	5	3,6875	-1,4375	0	0		
2,9375	3,3125	1,9375	0,5625	-2,125	0	0	0	0	0	0		
2,9375	3,3125	1,9375	0,5625	-2,125	-2,3125	0	0	0	0	0		
3,375	3,25	1,1875	1,5625	-0,625	-0,5625	5	3,6875	-1,4375	-0,625	0		
3,375	3,25	1,1875	1,5625	-0,625	-0,5625	5	3,6875	-1,4375	-0,625	0,9375		
3,25	1,1875	1,5625	-0,625	-0,5625	5	3,6875	-1,4375	-0,625	0,9375	0,9375		
2,9375	3,3125	1,9375	0,5625	-2,125	-2,3125	-0,5625	0	0	0	0		
2,9375	3,3125	1,9375	0,5625	-2,125	-2,3125	-0,5625	1,625	0	0	0		
2,9375	3,3125	1,9375	0,5625	-2,125	-2,3125	-0,5625	1,625	1,25	0	0		
1,1875	1,5625	-0,625	-0,5625	5	3,6875	-1,4375	-0,625	0,9375	0,9375	0,9375		
1,5625	-0,625	-0,5625	5	3,6875	-1,4375	-0,625	0,9375	0,9375	0,9375	1,25		
2,9375	3,3125	1,9375	0,5625	-2,125	-2,3125	-0,5625	1,625	1,25	1,875	0		
-0,625	-0,5625	5	3,6875	-1,4375	-0,625	0,9375	0,9375	0,9375	1,25	3,6875		
2,9375	3,3125	1,9375	0,5625	-2,125	-2,3125	-0,5625	1,625	1,25	1,875	4,625		
-0,5625	5	3,6875	-1,4375	-0,625	0,9375	0,9375	0,9375	1,25	3,6875	4,375		
5	3,6875	-1,4375	-0,625	0,9375	0,9375	0,9375	1,25	3,6875	4,375	3,9375		
3,6875	-1,4375	-0,625	0,9375	0,9375	0,9375	1,25	3,6875	4,375	3,9375	1,0625		
-1,4375	-0,625	0,9375	0,9375	0,9375	1,25	3,6875	4,375	3,9375	1,0625	-2,9375		
-0,625	0,9375	0,9375	0,9375	1,25	3,6875	4,375	3,9375	1,0625	-2,9375	-2,875		
1,875	3,375	0	0	0	0	0	0	0	0	0		
1,875	3,375	4,375	0	0	0	0	0	0	0	0		
1,875	3,375	4,375	1,4375	0	0	0	0	0	0	0		
3	0	0	0	0	0	0	0	0	0	0		
1,875	3,375	4,375	1,4375	-1,5	0	0	0	0	0	0		
1,875	3,375	4,375	1,4375	-1,5	-2,875	0	0	0	0	0		
3	0,5625	0	0	0	0	0	0	0	0	0		
3	0,5625	0,75	0	0	0	0	0	0	0	0		
3	0,5625	0,75	0,875	0	0	0	0	0	0	0		
3	0,5625	0,75	0,875	2,5	0	0	0	0	0	0		
3	0,5625	0,75	0,875	2,5	0,625	0	0	0	0	0		
3	0,5625	0,75	0,875	2,5	0,625	0,75	0	0	0	0		
3	0,5625	0,75	0,875	2,5	0,625	0,75	-1,3125	0	0	0		
3	0,5625	0,75	0,875	2,5	0,625	0,75	-1,3125	-0,75	0	0		
3	0,5625	0,75	0,875	2,5	0,625	0,75	-1,3125	-0,75	0,875	0		
3	0,5625	0,75	0,875	2,5	0,625	0,75	-1,3125	-0,75	0,875	-0,5625		
0,5625	0,75	0,875	2,5	0,625	0,75	-1,3125	-0,75	0,875	-0,5625	-0,75		
0,75	0,875	2,5	0,625	0,75	-1,3125	-0,75	0,875	-0,5625	-0,75	-2,6875		
0,875	2,5	0,625	0,75	-1,3125	-0,75	0,875	-0,5625	-0,75	-2,6875	2,875		
2,5	0,625	0,75	-1,3125	-0,75	0,875	-0,5625	-0,75	-2,6875	2,875	4		
0,625	0,75	-1,3125	-0,75	0,875	-0,5625	-0,75	-2,6875	2,875	4	3,1875		
1,875	3,375	4,375	1,4375	-1,5	-2,875	-1,625	0	0	0	0		
1,875	3,375	4,375	1,4375	-1,5	-2,875	-1,625	3,9375	0	0	0		
1,875	3,375	4,375	1,4375	-1,5	-2,875	-1,625	3,9375	3,875	0	0		
1,875	3,375	4,375	1,4375	-1,5	-2,875	-1,625	3,9375	3,875	3,1875	0		
1,875	3,375	4,375	1,4375	-1,5	-2,875	-1,625	3,9375	3,875	3,1875	3		
3,375	4,375	1,4375	-1,5	-2,875	-1,625	3,9375	3,875	3,1875	3	2,6875		
4,375	1,4375	-1,5	-2,875	-1,625	3,9375	3,875	3,1875	3	2,6875	1,75		
1,4375	-1,5	-2,875	-1,625	3,9375	3,875	3,1875	3	2,6875	1,75	-1,1875		
0,75	-1,3125	-0,75	0,875	-0,5625	-0,75	-2,6875	2,875	4	3,1875	1,4375		
-1,3125	-0,75	0,875	-0,5625	-0,75	-2,6875	2,875	4	3,1875	1,4375	-0,5625		
-1,5	-2,875	-1,625	3,9375	3,875	3,1875	3	2,6875	1,75	-1,1875	-3,25		
-2,875	-1,625	3,9375	3,875	3,1875	3	2,6875	1,75	-1,1875	-3,25	-4,1875		
-1,625	3,9375	3,875	3,1875	3	2,6875	1,75	-1,1875	-3,25	-4,1875	-2,625		
3,9375	3,875	3,1875	3	2,6875	1,75	-1,1875	-3,25	-4,1875	-2,625	-2,3125		
-0,75	0,875	-0,5625	-0,75	-2,6875	2,875	4	3,1875	1,4375	-0,5625	-2,125		
0,875	-0,5625	-0,75	-2,6875	2,875	4	3,1875	1,4375	-0,5625	-2,125	-2,25		
3,875	3,1875	3	2,6875	1,75	-1,1875	-3,25	-4,1875	-2,625	-2,3125	-2,875		
-0,5625	-0,75	-2,6875	2,875	4	3,1875	1,4375	-0,5625	-2,125	-2,25	-2,4375		
-0,75	-2,6875	2,875	4	3,1875	1,4375	-0,5625	-2,125	-2,25	-2,4375	-1,5625		
-2,6875	2,875	4	3,1875	1,4375	-0,5625	-2,125	-2,25	-2,4375	-1,5625	2,5		
2,875	4	3,1875	1,4375	-0,5625	-2,125	-2,25	-2,4375	-1,5625	2,5	1,4375		
4	3,1875	1,4375	-0,5625	-2,125	-2,25	-2,4375	-1,5625	2,5	1,4375	2,5		
3,1875	1,4375	-0,5625	-2,125	-2,25	-2,4375	-1,5625	2,5	1,4375	2,5	2,125		
1,4375	-0,5625	-2,125	-2,25	-2,4375	-1,5625	2,5	1,4375	2,5	2,125	2,625		
-0,5625	-2,125	-2,25	-2,4375	-1,5625	2,5	1,4375	2,5	2,125	2,625	3,9375		
3,1875	3	2,6875	1,75	-1,1875	-3,25	-4,1875	-2,625	-2,3125	-2,875	4,3125		
3	2,6875	1,75	-1,1875	-3,25	-4,1875	-2,625	-2,3125	-2,875	4,3125	5,0625		
-2,125	-2,25	-2,4375	-1,5625	2,5	1,4375	2,5	2,125	2,625	3,9375	2,9375		
-2,25	-2,4375	-1,5625	2,5	1,4375	2,5	2,125	2,625	3,9375	2,9375	2,25		
2,6875	1,75	-1,1875	-3,25	-4,1875	-2,625	-2,3125	-2,875	4,3125	5,0625	3,8125		
1,75	-1,1875	-3,25	-4,1875	-2,625	-2,3125	-2,875	4,3125	5,0625	3,8125	3		
-2,4375	-1,5625	2,5	1,4375	2,5	2,125	2,625	3,9375	2,9375	2,25	-6		
-1,5625	2,5	1,4375	2,5	2,125	2,625	3,9375	2,9375	2,25	-6	-3,8125		
-1,1875	-3,25	-4,1875	-2,625	-2,3125	-2,875	4,3125	5,0625	3,8125	3	-1,4375		
2,5	1,4375	2,5	2,125	2,625	3,9375	2,9375	2,25	-6	-3,8125	-6,125		
-3,25	-4,1875	-2,625	-2,3125	-2,875	4,3125	5,0625	3,8125	3	-1,4375	-3,875		
-4,1875	-2,625	-2,3125	-2,875	4,3125	5,0625	3,8125	3	-1,4375	-3,875	-4,6875		
-2,625	-2,3125	-2,875	4,3125	5,0625	3,8125	3	-1,4375	-3,875	-4,6875	-3,3125		
-2,3125	-2,875	4,3125	5,0625	3,8125	3	-1,4375	-3,875	-4,6875	-3,3125	-3,125		
1,8125	0	0	0	0	0	0	0	0	0	0		
1,8125	2,1875	0	0	0	0	0	0	0	0	0		
1,8125	2,1875	1,9375	0	0	0	0	0	0	0	0		
1,8125	2,1875	1,9375	4,9375	0	0	0	0	0	0	0		
1,8125	2,1875	1,9375	4,9375	3,9375	0	0	0	0	0	0		
1,8125	2,1875	1,9375	4,9375	3,9375	2,875	0	0	0	0	0		
1,8125	2,1875	1,9375	4,9375	3,9375	2,875	3	0	0	0	0		
1,8125	2,1875	1,9375	4,9375	3,9375	2,875	3	3,3125	0	0	0		
1,8125	2,1875	1,9375	4,9375	3,9375	2,875	3	3,3125	3,5	0	0		
3,5	0	0	0	0	0	0	0	0	0	0		
1,8125	2,1875	1,9375	4,9375	3,9375	2,875	3	3,3125	3,5	0,5625	0		
3,5	3,3125	0	0	0	0	0	0	0	0	0		
3,5	3,3125	3,8125	0	0	0	0	0	0	0	0		

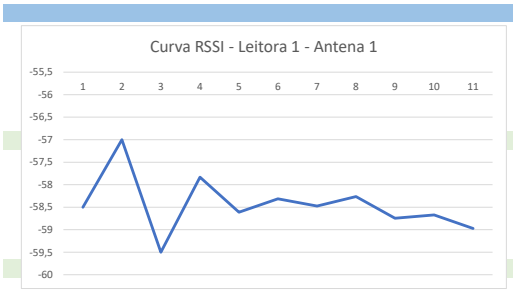
3,5	3,3125	3,8125	1,75	0	0	0	0	0	0	0		
1,8125	2,1875	1,9375	4,9375	3,9375	2,875	3	3,3125	3,5	0,5625	-2,1875		
2,1875	1,9375	4,9375	3,9375	2,875	3	3,3125	3,5	0,5625	-2,1875	-2		
3,5	3,3125	3,8125	1,75	-1,75	0	0	0	0	0	0		
3,5	3,3125	3,8125	1,75	-1,75	-0,9375	0	0	0	0	0		
3,5	3,3125	3,8125	1,75	-1,75	-0,9375	1,25	0	0	0	0		
3,5	3,3125	3,8125	1,75	-1,75	-0,9375	1,25	0,8125	0	0	0		
3,5	3,3125	3,8125	1,75	-1,75	-0,9375	1,25	0,8125	1,3125	0	0		
3,5	3,3125	3,8125	1,75	-1,75	-0,9375	1,25	0,8125	1,3125	1,875	0		
3,5	3,3125	3,8125	1,75	-1,75	-0,9375	1,25	0,8125	1,3125	1,875	4,0625		
3,3125	3,8125	1,75	-1,75	-0,9375	1,25	0,8125	1,3125	1,875	4,0625	4,625		
3,8125	1,75	-1,75	-0,9375	1,25	0,8125	1,3125	1,875	4,0625	4,625	-3,8125		
1,75	-1,75	-0,9375	1,25	0,8125	1,3125	1,875	4,0625	4,625	-3,8125	-3,5		
1,9375	4,9375	3,9375	2,875	3	3,3125	3,5	0,5625	-2,1875	-2	-1,8125		
4,9375	3,9375	2,875	3	3,3125	3,5	0,5625	-2,1875	-2	-1,8125	-0,875		
3,9375	2,875	3	3,3125	3,5	0,5625	-2,1875	-2	-1,8125	-0,875	-3		
2,875	3	3,3125	3,5	0,5625	-2,1875	-2	-1,8125	-0,875	-3	-5,25		
3	3,3125	3,5	0,5625	-2,1875	-2	-1,8125	-0,875	-3	-5,25	-2,5625		
-2,875	4,3125	5,0625	3,8125	3	-1,4375	-3,875	-4,6875	-3,3125	-3,125	-0,875		
4,3125	5,0625	3,8125	3	-1,4375	-3,875	-4,6875	-3,3125	-3,125	-0,875	1,5625		
0,9375	0,9375	0,9375	1,25	3,6875	4,375	3,9375	1,0625	-2,9375	-2,875	2,125		
0,9375	0,9375	1,25	3,6875	4,375	3,9375	1,0625	-2,9375	-2,875	2,125	-2,625		
0,9375	1,25	3,6875	4,375	3,9375	1,0625	-2,9375	-2,875	2,125	-2,625	-0,875		
3,3125	1,9375	0,5625	-2,125	-2,3125	-0,5625	1,625	1,25	1,875	4,625	-5,375		

Tempo Doppler												
0	0	0	0	0	0	0	0	0	0	0		
0	55,4012	0	0	0	0	0	0	0	0	0		
61,4164	0	0	0	0	0	0	0	0	0	0		1
61,4164	241,9069	0	0	0	0	0	0	0	0	0		
0	55,4012	244,7878	0	0	0	0	0	0	0	0		
0	55,4012	244,7878	741,4085	0	0	0	0	0	0	0		
61,4164	241,9069	991,0493	0	0	0	0	0	0	0	0		
61,4164	241,9069	991,0493	992,2347	0	0	0	0	0	0	0		
61,4164	241,9069	991,0493	992,2347	1242,7769	0	0	0	0	0	0		
61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	0	0	0	0	0		
4943,0372	0	0	0	0	0	0	0	0	0	0		
61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	0	0	0	0		
61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	0	0	0		
61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	0	0		
0	55,4012	244,7878	741,4085	10991,6913	0	0	0	0	0	0		
0	55,4012	244,7878	741,4085	10991,6913	10993,0574	0	0	0	0	0		
61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	0		
61,4164	241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612		
241,9069	991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083		
0	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	0	0	0	0		
0	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	16242,8737	0	0	0		
0	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	16242,8737	16741,5382	0	0		
991,0493	992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056		
992,2347	1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085		
0	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	16242,8737	16741,5382	16995,8566	0		
1242,7769	1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678		
0	55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	16242,8737	16741,5382	16995,8566	17742,8505		
1492,1545	10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478		
10241,2851	10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653		
10243,3632	10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549		
10744,9541	13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	18491,2526		
13490,7682	14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	18491,2526	18741,3122		
4943,0372	27443,6946	0	0	0	0	0	0	0	0	0		
4943,0372	27443,6946	27444,9933	0	0	0	0	0	0	0	0		
4943,0372	27443,6946	27444,9933	27694,5975	0	0	0	0	0	0	0		
27696,3162	0	0	0	0	0	0	0	0	0	0		
4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	0	0	0	0	0	0		
4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	0	0	0	0	0		
27696,3162	33943,9055	0	0	0	0	0	0	0	0	0		
27696,3162	33943,9055	33945,2577	0	0	0	0	0	0	0	0		
27696,3162	33943,9055	33945,2577	34193,5997	0	0	0	0	0	0	0		
27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	0	0	0	0	0	0		
27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	0	0	0	0	0		
27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	0	0	0	0		
27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	0	0	0		
27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	0	0		
27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	0		
27696,3162	33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061		
33943,9055	33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157		
33945,2577	34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977		
34193,5997	34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543		
34444,4887	35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419		
35943,7055	36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332		
4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	0	0	0	0		
4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	0	0	0		
4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	0	0		
4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	0		
4943,0372	27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672		
27443,6946	27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389		
27444,9933	27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997		
27694,5975	27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119		
36445,0185	37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343		
37194	37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101		
27944,7275	27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884		
27949,9395	47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266		
47944,6178	53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916		
53944,6342	53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806		
37443,7157	38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052		
38694,5455	40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549		
53948,3669	53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883		
40694,6061	42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829		
42943,9157	44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236		
44444,0977	47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921		
47194,1543	47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836		
47444,0419	47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138		
47446,332	54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217		
54696,4343	54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028		
54706,6101	54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874		
53958,5823	54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272		
54194,672	54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621		
54952,5052	55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	60225,177		
55194,3549	55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	60225,177	60241,3146		
54198,2389	54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785		
54444,7997	54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904		
55444,2829	55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	60225,177	60241,3146	60456,8282		
55445,2236	59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	60225,177	60241,3146	60456,8282	60459,7947		
54694,5119	54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516		
59194,5921	59444,3836	59695,0138	59697,217	59703,8028	59944,2874	60225,177	60241,3146	60456,8282	60459,7947	60696,1124		
54709,5884	54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138		
54944,266	54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	60945,4313		
54945,5916	54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	60945,4313	60950,3092		
54947,3806	55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	60945,4313	60950,3092	60951,6426		
66865,6655	0	0	0	0	0	0	0	0	0	0		
66865,6655	67116,0371	0	0	0	0	0	0	0	0	0		
66865,6655	67116,0371	67118,1527	0	0	0	0	0	0	0	0		
66865,6655	67116,0371	67118,1527	67365,9737	0	0	0	0	0	0	0		
66865,6655	67116,0371	67118,1527	67365,9737	67368,596	0	0	0	0	0	0		
66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	0	0	0	0	0		
66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	0	0	0	0		
66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	0	0	0		
66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	0	0		
67881,5174	0	0	0	0	0	0	0	0	0	0		
66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	0		
67881,5174	68366,1905	0	0	0	0	0	0	0	0	0		
67881,5174	68366,1905	68370,7653	0	0	0	0	0	0	0	0		

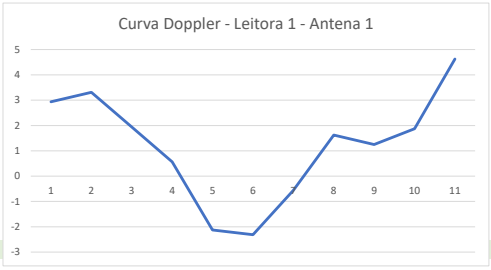
67881,5174	68366,1905	68370,7653	68615,9201	0	0	0	0	0	0	0		
66865,6655	67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295		
67116,0371	67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902		
67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	0	0	0	0	0	0		
67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	0	0	0	0	0		
67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	0	0	0	0		
67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	0	0	0		
67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	0	0		
67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	73619,0375	0		
67881,5174	68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	73619,0375	73866,0067		
68366,1905	68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	73619,0375	73866,0067	73867,0761		
68370,7653	68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	73619,0375	73866,0067	73867,0761	74116,2457		
68615,9201	69366,6204	69368,8492	73116,2631	73127,4421	73616,0064	73619,0375	73866,0067	73867,0761	74116,2457	74366,9877		
67118,1527	67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	74618,3672		
67365,9737	67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	74618,3672	74627,7978		
67368,596	67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	74618,3672	74627,7978	74866,3177		
67616,4866	67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	74618,3672	74627,7978	74866,3177	75367,2235		
67622,9438	67866,3463	67877,3397	68115,4739	68622,5295	68633,4902	74618,3672	74627,7978	74866,3177	75367,2235	75372,1055		
55195,2883	60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	60945,4313	60950,3092	60951,6426	75696,2426		
60195,272	60207,3621	60444,785	60450,6904	60694,6516	60944,4138	60945,4313	60950,3092	60951,6426	75696,2426	75944,9133		
14494,2612	14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	18491,2526	18741,3122	82992,172		
14991,4083	16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	18491,2526	18741,3122	82992,172	83241,8603		
16991,6056	16993,3085	17491,6678	17991,0478	17992,1653	18241,3549	18491,2526	18741,3122	82992,172	83241,8603	83491,4789		
55,4012	244,7878	741,4085	10991,6913	10993,0574	15991,7046	16242,8737	16741,5382	16995,8566	17742,8505	83741,3075		

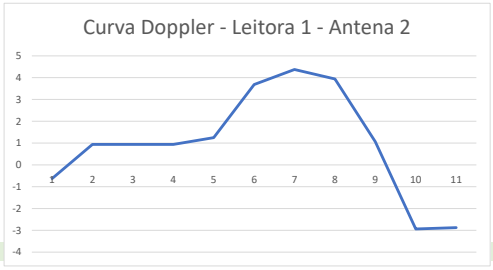
Ambiente-LAST-RSSI-PEAK-TIME	Ambiente-RSSI-Last-value	Ambiente-RSSI-Mean	Ambiente-RSSI-Median	Ambiente-Doppler	DopplerAndRSSI
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)	Area_Externa(0)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)
Corredor_Baias(3)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)

Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Reuniao(2)	Sala_Reuniao(2)	Sala_Reuniao(2)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Corredor_Baias(3)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)	Sala_Principal(1)
Area_Externa(0)	Area_Externa(0)	Sala_Principal(1)	Sala_Principal(1)	Area_Externa(0)	Area_Externa(0)


Curva RSSI - Leitora 1 - Antena 1
-55,5
-56
-56,5
-57
-57,5
-58
-58,5
-59
-59,5
-60
1 2 3 4 5 6 7 8 9 10 11


Título do Gráfico
-56
-56,5
-57
-57,5
-58
-58,5
-59
1 2 3 4 5 6 7 8 9 10 11


Curva Doppler - Leitora 1 - Antena 1
5
4
3
2
1
0
-1
-2
-3
1 2 3 4 5 6 8 9 10 11


Curva Doppler - Leitora 1 - Antena 2
5
4
3
2
1
0
-1
-2
-3
-4
1 2 3 4 5 6 7 8 9 10 11