

Universidade Federal de Catalão
Instituto de Biotecnologia
Curso de Bacharelado em Ciência da Computação

Protocolo de Transmissão por Proximidade com Dados sobre Som

Marcelo Ribeiro Tormim

Catalão – GO
2023

Marcelo Ribeiro Tormim

Protocolo de Transmissão por Proximidade com Dados sobre Som

Monografia apresentada ao Curso de Bacharelado em Ciência da Computação da Universidade Federal de Catalão, como parte dos requisitos para obtenção do grau de Bacharel em Ciência da Computação. *VERSÃO REVISADA*

Orientador: Prof. Dr. Ricardo Couto Antunes da Rocha

Catalão – GO
2023

Ficha de identificação da obra elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UFCAT.

Tormim, Marcelo Ribeiro

Protocolo de Transmissão por Proximidade com Dados sobre
Som [manuscrito] / Marcelo Ribeiro Tormim. – 2023.

66 p.: il.

Orientador: Prof. Dr. Ricardo Couto Antunes da Rocha
Monografia (Graduação) – Universidade Federal de Catalão,
Instituto de Biotecnologia, Ciência da Computação, 2023.
Bibliografia.

1. Dados sobre Som. 2. Sistemas Distribuídos. 3. Redes de
Computadores. I. Rocha, Ricardo Couto Antunes da, orient. II.
Título.

CDU 004

Marcelo Ribeiro Tormim

Protocolo de Transmissão por Proximidade com Dados sobre Som

Monografia apresentada ao curso de Bacharelado em Ciência da Computação da Universidade Federal de Catalão.

Trabalho aprovado em 24 de Agosto de 2023.

Ricardo Couto Antunes da Rocha
Orientador

Tércio Alberto dos Santos Filho
Universidade Federal de Catalão

Sérgio Francisco da Silva
Universidade Federal de Catalão

Catalão – GO
2023

RESUMO

TORMIM, M. R.. **Protocolo de Transmissão por Proximidade com Dados sobre Som.** 2023. [66](#) p. Monografia (Graduação) – Instituto de Biotecnologia, Universidade Federal de Catalão, Catalão – GO.

Dados sobre Som é um mecanismo para comunicação entre dispositivos através de informação codificada em forma de ondas sonoras. Suas características particulares possibilitam um meio de comunicação sem fio um-para-muitos sem necessidade de pareamento ou hardware de rede adicional, sendo um método atraente para fácil transmissão de pequenas quantidades de dados entre dispositivos próximos em comparação a outras soluções disponíveis. O estabelecimento de um protocolo para comunicação entre dispositivos utilizando Dados sobre Som possibilita o aproveitamento desse método de transmissão de dados em aplicações diversas. Um protocolo de descoberta de dispositivos determina como dispositivos se anunciam e são percebidos pelos outros em um ambiente compartilhado. Um protocolo de transporte de dados permite o envio de bytes entre dispositivos previamente reconhecidos. Esta monografia desenvolveu um protocolo de descoberta de dispositivos e transporte de dados fazendo uso de Dados sobre Som. Ele opera na camada de enlace e é construído sobre o protocolo de camada física oferecido pela biblioteca gwave. Experimentos foram realizados para avaliação de seu desempenho em termos do efeito de distância ou obstrução/interferência do sinal na qualidade da comunicação. Transmissões inaudíveis se mostraram eficazes em distâncias de até três metros na presença de ruído moderado, indicando potencial para emprego em cenários com comunicação de curto alcance em ambientes fechados. Testes mais extensos são necessários para maior compreensão do desempenho possível em cenários reais de uso, bem como desenvolvimento adicional para resolver limitações do protocolo, incluindo a ausência de um método para redução de colisões causadas por transmissões simultâneas.

Palavras-chave: Dados sobre Som, Sistemas Distribuídos, Redes de Computadores.

ABSTRACT

TORMIM, M. R.. **Protocolo de Transmissão por Proximidade com Dados sobre Som.** 2023.
66 p. Monografia (Graduação) – Instituto de Biotecnologia, Universidade Federal de Catalão,
Catalão – GO.

Data over Sound is a method for device communication that encodes data in the form of sound waves. It enables one-to-many wireless communication without pairing or additional network hardware, allowing easy transmission of small amounts of data between nearby devices compared to other available solutions. The development of a communication protocol for Data over Sound is necessary to leverage it in various applications. A device discovery protocol specifies how devices advertise themselves so that other devices can perceive them in a shared environment. A data transport protocol enables the sending of bytes between previously recognized devices. This work developed a device discovery and data transport protocol for Data over Sound. The proposed protocol is part of the link layer and uses ggwave library as an implementation of the physical layer. Our evaluation tests showed the effect distance or signal obstruction/interference has on its performance in terms of communication quality. Inaudible transmissions proved effective over distances of up to three meters in the presence of moderate noise, indicating potential for use indoors in short-range communication scenarios. Further testing is necessary for a comprehensive understanding of the performance in real-usage scenarios and to address the protocol limitations, including the absence of a method to reduce collisions caused by simultaneous transmissions.

Keywords: Data over Sound, Distributed Systems, Computer Networks.

SUMÁRIO

| | | |
|----------|---|-----------|
| 1 | INTRODUÇÃO | 11 |
| 1.1 | Objetivo | 12 |
| 1.2 | Organização da Monografia | 12 |
| 2 | TRANSMISSÃO EM DADOS SOBRE SOM | 13 |
| 2.1 | Dados sobre som | 13 |
| 2.2 | Cenário de Uso | 15 |
| 2.3 | Vantagens e Limitações | 17 |
| 2.4 | Hardware para Transmissão e Recepção | 17 |
| 3 | PROTOCOLOS DOS DE CAMADA FÍSICA | 19 |
| 3.1 | Quiet | 19 |
| 3.2 | AstroMech | 19 |
| 3.3 | SoniTalk | 20 |
| 3.4 | ggwave | 20 |
| 4 | PROTOCOLO GGWAVE PARA DADOS SOBRE SOM | 21 |
| 4.1 | Transmissão e Camada Física | 21 |
| 4.2 | Correção de erros | 27 |
| 4.3 | Exemplo | 30 |
| 5 | EXPERIMENTOS COM GGWAVE | 33 |
| 5.1 | Ambiente de Experimentação | 33 |
| 5.2 | Efeito de Distância na Transmissão | 35 |
| 5.3 | Influência de Ruídos no Ambiente | 36 |
| 5.4 | Interferências entre Transmissões Simultâneas | 38 |
| 5.5 | Influência de Obstáculos Físicos | 39 |
| 5.6 | Conclusão | 43 |
| 6 | PROTOCOLO DE TRANSMISSÃO POR PROXIMIDADE | 45 |
| 6.1 | Cenário | 45 |
| 6.2 | Transmissão por Proximidade no Bluetooth | 46 |
| 6.2.1 | <i>Bluetooth Low Energy</i> | 46 |
| 6.2.2 | <i>Protocolo de Advertisement</i> | 46 |

| | | |
|--------------------|---|----|
| 6.2.3 | <i>Protocolo de Estabelecimento de Conexão</i> | 47 |
| 6.2.4 | <i>Análise Comparativa e de Viabilidade entre Bluetooth e Dados sobre Som</i> | 48 |
| 6.3 | Projeto do Protocolo | 48 |
| 6.3.1 | <i>Endereçamento</i> | 49 |
| 6.3.2 | <i>Estrutura de Mensagens</i> | 49 |
| 6.3.3 | <i>Processamento de Mensagens</i> | 50 |
| 6.3.4 | <i>Implementação</i> | 50 |
| 6.4 | Avaliação | 51 |
| 6.4.1 | <i>Ambiente de Experimentação</i> | 51 |
| 6.4.2 | <i>Interferências</i> | 52 |
| 6.4.3 | <i>Discussão</i> | 52 |
| 6.5 | Conclusão | 53 |
| 7 | CONCLUSÃO | 55 |
| 7.1 | Contribuições | 55 |
| 7.2 | Trabalhos Futuros | 56 |
| REFERÊNCIAS | | 59 |
| APÊNDICE A | CÓDIGO DE TRANSMISSÃO E RECEPÇÃO DE MENSAGENS | 63 |



INTRODUÇÃO

A tecnologia de dados sobre som (*data-over-sound* - DoS) é um mecanismo para comunicação entre dispositivos através de informação codificada em forma de ondas sonoras. Para a codificação de dados, podem ser empregadas frequências abaixo de 15 kHz, que serão audíveis, ou frequências a partir de 15 kHz até usualmente 20 kHz, que estão além da faixa de audição comum para humanos adultos (ASHIHARA, 2007).

Passos do processo de comunicação incluem a codificação por um dispositivo de dados na forma de som, reprodução por um alto-falante do som resultante, captação dessa transmissão por um microfone e decodificação do som por um dispositivo para recuperar a informação. Dependendo de variações entre situações de uso e implementações específicas, a tecnologia é capaz de transmitir entre 50 a 100 bits por segundo (BUBLEY, 2017).

As características particulares de DoS possibilitam um meio de comunicação sem fio um-para-muitos sem necessidade de pareamento ou hardware de rede adicional, sendo um método atraente para fácil transmissão de pequenas quantidades de dados entre dispositivos próximos quando comparado a outras soluções disponíveis (MARNEWICK *et al.*, 2019). O estabelecimento de um protocolo para comunicação entre dispositivos utilizando dados sobre som possibilita o aproveitamento desse método de transmissão de dados em aplicações diversas.

Um protocolo de descoberta de dispositivos determina como os dispositivos se anunciam e são percebidos por outros dispositivos em um ambiente compartilhado. Protocolos como *Bluetooth Low Energy* (BLE) implementam mecanismos de descoberta de dispositivos por meio de envio de anúncios periódicos que podem ser detectados por dispositivos próximos (LIU *et al.*, 2013). Um protocolo de transporte de dados é o protocolo que de fato permite o envio de bytes entre dispositivos que foram previamente reconhecidos.

1.1 Objetivo

O objetivo deste trabalho foi desenvolver um protocolo de descoberta de dispositivos e transporte de dados que faça uso de dados sobre som.

A compatibilidade de soluções preexistentes de transmissão de dados por som com as intenções do trabalho foi analisada, para determinar a viabilidade de seu emprego completo ou parcialmente em substituição ao desenvolvimento de soluções próprias durante o processo, projetadas especificamente para atender as necessidades do trabalho.

Os objetivos específicos deste trabalho, derivados do objetivo principal, foram:

- Projetar e desenvolver um protocolo de descoberta de dispositivos e transporte de dados utilizando *data-over-sound*.
- Avaliar o desempenho do protocolo em termos de efetividade em situações não ideais (interferência e ocultação de sinal) e efeito de distância na qualidade da comunicação.

1.2 Organização da Monografia

Esta monografia está organizada em sete capítulos:

- **Capítulo 1:** Introduz o tema abordado, bem como estabelece os objetivos principal e específicos do trabalho.
- **Capítulo 2:** Apresenta informações sobre a tecnologia dados sobre som de transmissão de dados, suas características, aplicações, vantagens/limitações e considerações associadas ao hardware necessário.
- **Capítulo 3:** Apresenta os protocolos de comunicação da tecnologia dados sobre som que estudamos em pesquisa para a realização deste trabalho.
- **Capítulo 4:** Descreve detalhes técnicos de funcionamento do ggwave, que escolhemos como solução da camada física sobre a qual nosso protocolo DoS, de atuação na camada de enlace, é construído.
- **Capítulo 5:** Detalha os experimentos que realizamos com o ggwave para avaliar o seu desempenho.
- **Capítulo 6:** Apresenta o projeto de nosso protocolo DoS de transmissão por proximidade, detalhes de sua implementação e o experimento que realizamos para avaliá-lo.
- **Capítulo 7:** Conclui a monografia, com exposição de nossas considerações a respeito dos resultados obtidos e sobre potenciais trabalhos futuros.



TRANSMISSÃO EM DADOS SOBRE SOM

Este capítulo apresenta informações sobre a tecnologia *data-over-sound* (DoS) de transmissão de dados, suas características particulares, aplicações, vantagens/limitações e considerações associadas ao hardware necessário.

2.1 Dados sobre som

A tecnologia de dados sobre som consiste na codificação de dados sob a forma de sinais de áudio como um mecanismo para comunicação entre dispositivos de informação codificada em forma de áudio. Implementações de dados sobre som podem empregar frequências abaixo de 15 kHz, que serão audíveis, ou frequências a partir de 15 kHz até usualmente 20 kHz, além da faixa de audição comum para humanos adultos. O processo de comunicação envolve a codificação dos dados na forma de som por um dispositivo, reprodução por um alto-falante do áudio gerado, captação da transmissão por um microfone e sua decodificação por um dispositivo para recuperar a informação. Dependendo dos detalhes de cada implementação, características do ambiente e proximidade entre receptor/transmissor, a tecnologia é capaz de transmitir entre 50 a 100 bits por segundo.

Representação de som em formato digital é necessária para as operações de processamento de sinal associadas à transmissão/recepção de dados sobre som. Isso envolve a codificação da onda sonora do sinal de áudio como uma sequência contínua de amostras numéricas representando sua amplitude em um momento específico. A faixa de frequências sonoras entre 20 e 20.000 Hz é amplamente aceita como os limites da audição humana (ROSEN; HOWELL, 2011). Para ser capaz de reproduzir certa faixa de frequências, áudio digital precisa de uma taxa de amostragem de pelo menos duas vezes a frequência máxima do sinal gravado, segundo estabelecido pelo teorema de amostragem Nyquist-Shannon. Baseando-se nesse fato, as taxas de amostragem de 44,1 kHz e 48 kHz são os padrões comumente utilizados em aplicações que visam reprodução completa da faixa audível e o alvo geral de fidelidade almejado por equipamentos

convencionais de captura/reprodução de áudio.

A Análise de Fourier é o estudo da forma como funções gerais podem ser representadas por somas de funções trigonométricas mais simples, sendo um elemento fundamental para diversas técnicas de processamento de sinal e para o funcionamento da tecnologia DoS. As variantes da análise de Fourier podem ser divididas em categorias de acordo com o tipo de sinal sendo considerado, o qual pode ser periódico ou aperiódico e contínuo ou discreto. Uma série de Fourier é uma soma que representa uma função periódica como uma soma de ondas senoidais cujas frequências são múltiplos inteiros da frequência mais baixa dessa função. Funções aperiódicas não podem ser representadas por série de Fourier. Entretanto, uma extensão da série de Fourier chamada transformada de Fourier, que trata tais funções como funções periódicas de período infinito, pode ser utilizada. Dessa forma, uma transformada de Fourier é capaz de decompor funções dependendo de tempo (variação do sinal ao longo do tempo) em funções dependendo de frequência (quanto do sinal reside em cada faixa de frequência). Para lidar com sinais discretos existem as chamadas transformada discreta de Fourier, utilizada com sinais periódicos, e transformada de Fourier em tempo discreto, utilizada com sinais aperiódicos.

Considerando que a transformada de Fourier envolve sinais de comprimento infinito, para lidar com um número finito de amostras de um sinal é necessário tratá-las como parte de uma série de amostras que se estendem do infinito negativo ao infinito positivo. Tais amostras imaginárias podem consistir de repetições das amostras finitas disponíveis, gerando um sinal aparentemente periódico (cujo período é igual ao número de amostras) passível a aplicação de transformada discreta de Fourier, ou podem todas ter valor zero, gerando um sinal aparentemente aperiódico passível a aplicação de transformada de Fourier em tempo discreto.

Computadores só são capazes de manipular dados de natureza discreta e comprimento finito. Para produzir um sinal aperiódico é necessário um número infinito de senoidais, tornando impossível o cálculo de transformada de Fourier em tempo discreto por meio de um algoritmo de computador. Dessa forma, a única variante de análise de Fourier possível em computadores é a transformada discreta de Fourier.

A transformação discreta de Fourier pode ser calculada de diversas formas, entretanto muitas das abordagem possíveis não são conducentes a criação de algoritmos eficientes apropriados para aplicações práticas. A *fast Fourier transform* (FFT) é um método altamente eficiente capaz de produzir resultados com tempos de computação centenas de vezes menores que as demais técnicas. Devido a isso, FFT é a abordagem empregada em virtualmente todas as aplicações práticas de processamento digital de sinais.

As ondas sonoras possuem características particulares que tornam a transmissão de dados sobre som, apesar de sua aparente pouca sofisticação e baixa velocidade, uma alternativa potencialmente atraente quando comparada a soluções convencionais empregando ondas de rádio ([MARNEWICK et al., 2019](#)).

O uso de som para transmissão de dados em situações envolvendo comunicação de curto alcance pode oferecer uma experiência descomplicada e eficaz. Abordagens comumente empregadas em tais situações têm problemas que as tornam menos desejáveis. Certas tecnologias (*near-field communication, quick-response code*) têm limitações diversas de acessibilidade: só comunicação de um sentido é permitida, linha de visão entre os dispositivos é necessária, comunicação só é possível com muita proximidade, entre outros. Os desafios listados não se aplicam à comunicação por som.

Considerando o emprego da tecnologia de dados sobre som, dispositivos não necessitam de funcionalidade/hardware adicional para suportá-la (apenas alto-falante e microfone convencionais), permitindo suporte fácil a dispositivos legado. O sistema operacional rodando nos dispositivos não afeta o funcionamento da tecnologia, transmissões podem ocorrer a certa distância (alguns metros) e comunicação dois sentidos é possível.

Outra situação em que a tecnologia DoS se destaca são ambientes sensíveis ao uso de rádio frequência (RF). Isso inclui UTIs de hospitais, certos laboratórios científicos, locais com estruturas metálicas que dificultam a propagação de sinais de rádio, entre outros (PERIYASAM; DHANASEKARAN, 2013). As ondas sonoras utilizadas para transmissão são ondas mecânicas, diferentemente das ondas eletromagnéticas de tecnologias de rádio transmissão como Wi-Fi e Bluetooth. Isso as tornam seguras em tais ambientes e especialmente úteis em situações envolvendo elementos móveis, em que conexões cabeadas não são possíveis.

Certas características específicas de ondas sonoras conferem algumas particularidades à tecnologia DoS que a tornam especialmente útil em certas aplicações e inviabilizam seu uso em outras. Diferentemente das ondas de rádios usadas em outras soluções, ondas sonoras (especialmente as de maior frequência) têm propagação limitada através de obstáculos físicos como paredes, restringindo comunicação apenas ao interior de uma sala, e uma velocidade de propagação muitas vezes inferior, tornando-as consideravelmente mais susceptíveis às consequências indesejadas de ecos e do efeito Doppler (GERASIMOV; BENDER, 2000).

A transmissão de dados via som, assim como outros métodos de comunicação sem fio, requer um conjunto de protocolos estabelecendo as regras que governam os diversos aspectos envolvidos no processo de comunicação, incluindo sinalização, correção de erro e demais elementos necessário para o funcionamento do sistema. As características específicas do ambiente e dos dispositivos almejados precisam guiar escolhas, associadas ao balanceamento entre taxa de dados e robustez da comunicação empregado por uma implementação de DoS, para garantir desempenho adequado.

2.2 Cenário de Uso

A transmissão de dados por som pode ser empregada em aplicações diversas, sendo especialmente atraentes em situações onde outros métodos apresentam limitações.

As características particularidades de ondas sonoras tornam a tecnologia DoS uma alternativa ou complemento atraente para aplicações envolvendo transmissões de baixa taxa de dados entre dispositivos próximos em ambientes internos. Comunicação do tipo ponto-a-ponto ou ponto-a-multiponto são oferecidas com baixos requerimentos de hardware e facilidade de uso. Aplicações envolvendo detecção de proximidade e presença em um ambiente também podem potencialmente se beneficiar do emprego de dados sobre som.

Existem inúmeros potenciais casos de uso envolvendo o emprego da tecnologia DoS nos mais diversos nichos. A seguir são citados alguns exemplos.

DoS pode ser empregada como uma solução de conectividade em ambientes industriais que, devido a implantação de processos de automação, passaram a necessitar de métodos para comunicação entre dispositivos. Em particular, ambientes industriais com restrições relacionadas à RF — locais com explosivos usados para mineração, locais com maquinário causador de interferência RF, entre outros — podem fazer uso de dados sobre som para transmissões de dados de curto alcance entre dispositivos, com emprego de sinais ultrassônicos em situações envolvendo presença de muito ruído mecânico para evitar interferência devido à sobreposição de frequências.

A realização de pagamentos via smartphones é uma área em que existem muitas experimentações envolvendo a tecnologia DoS. Aplicativos podem oferecer diretamente a funcionalidade em smartphone, sem dependências específicas relativas ao modelo do aparelho ou seu SO, necessitando meramente de acesso ao microfone e alto-falante do dispositivo. Essa é uma opção menos custosa e com mais potencial para ampla difusão que soluções tradicionais envolvendo chips de *near-field communication* (NFC).

Dadas as características particulares de ondas sonoras que limitam sua propagação além das paredes de uma sala, dados sobre som pode ser empregada para a detecção de presença e proximidade em um ambiente ([THIEL; KLOCH; LUKOWICZ, 2012](#); [SATOH *et al.*, 2013](#); [REN *et al.*, 2021](#)). Dispositivos podem realizar transmissões de tempos em tempos de modo a permitir que dispositivos próximos presentes no mesmo local o identifiquem.

Durante a pandemia global de coronavírus, aplicativos para rastreamento de contatos — registro das interações sociais de infectados com não-infectados para romper cadeias de contágio — ganharam destaque público dentre as iniciativas para combate do COVID-19. Em linhas gerais, o rastreamento digital de contatos pode ser descrito como um caso particular do problema de detecção de proximidade, envolvendo seu emprego como solução de identificação e acompanhamento para quebra de cadeias de transmissão. Dentre as estratégias empregadas nos protocolos das soluções disponíveis até o momento, BLE tem sido o método dominante de rastreamento de proximidade. Entretanto, existem preocupações em relação à sua eficácia, especialmente em relação à potencial imprecisão na detecção de situações de contato dado o uso da força de sinal como método para determinar a proximidade entre indivíduos. O emprego de ondas sonoras ultrassônicas para detecção de proximidade entre indivíduos em rastreamento

digital de contatos tem apresentado resultados promissores quando comparado a soluções empregando BLE. As características físicas de ondas sonoras ultrassônicas, como não propagação através de obstáculos físicos como paredes, permitem estimativas de distância entre pessoas consideravelmente mais precisas (LOH, 2020). Isso é alcançado a partir do princípio de tempo de voo (*time of flight* ou ToF), que envolve medição do tempo necessário para uma onda sonora viajar do dispositivo até um objeto e refletir de volta, informação que possibilita estimar a distância entre objeto e dispositivo com base em conhecimento científico à respeito da propagação de ondas sonoras.

2.3 Vantagens e Limitações

As características únicas associadas a transmissão de dados utilizando ondas sonoras tornam dados sobre som uma possibilidade atraente para diversas aplicações e, em contrapartida, inadequada em certas situações.

Dados sobre som é capaz de oferecer transmissões de baixa taxa de dados de baixo a médio alcance entre dispositivos que se comparam favoravelmente ao oferecido por tecnologias de conectividade alternativas disponíveis. Transmissões via som não requerem linha de visão entre dispositivos, permitem comunicação um-para-muitos e comunicação em dois sentidos, mantêm-se confinadas aos limites físicos de um espaço e funcionam em ambientes com restrições em relação à RF.

Os componentes de hardware necessários para possibilitar a implementação de comunicação por dados sobre som podem ser adquiridos com baixo custo ou dispositivos pré-existentes com entrada/saída de áudio podem ser empregados. Isso torna a tecnologia viável e acessível para aplicações diversas.

As limitações da tecnologia DoS envolvem a taxa de dados e alcance de suas transmissões, associadas à largura de banda disponível e características físicas do meio acústico. Dadas as restrições em relação a largura de faixas de frequência disponíveis para uso considerando limitações dos microfones/alto-falantes disponíveis e as dificuldades de propagação enfrentadas por ondas mecânicas como as ondas sonoras, alcance superior e taxas de dados mais altas são simplesmente inviáveis em situações reais (JOSEPH-RAZ, 2020).

2.4 Hardware para Transmissão e Recepção

Os requisitos fundamentais para possibilitar comunicação via som são um alto-falante para transmissão dos dados codificados na forma de sinais de áudio, um microfone para captação de áudio e um processador para realização das operações de codificação/decodificação necessárias. Tais requisitos são atendidos adequadamente por qualquer smartphone ou tablet e, com avanços tecnológicos recentes, soluções de baixo consumo de energia e baixo custo com

poder de processamento suficiente estão disponíveis. Apesar do grande número de dispositivos que podem ser empregados efetivamente em implementações de DoS, certas considerações adicionais são necessárias para determinar como as características dos componentes envolvidos afetam o funcionamento de uma certa implementação.

Microfones e alto-falantes convencionais apresentam certas limitações em relação a faixa de frequências que suportam e a precisão com que reproduzem essas frequências. Tais limitações, embora aceitáveis nas aplicações para as quais foram originalmente projetados, têm impacto sobre a performance possível em seu emprego para transmissão de dados via som.

Os microfones e alto-falantes disponíveis usualmente se dividem entre sistemas projetados para reprodução geral de sons, abrangendo o espectro de frequências audíveis por humanos (20 Hz a 20 kHz), e sistemas que cobrem uma porção reduzida do espectro audível visando uma aplicação específica, como as frequências suficientes para reprodução inteligível de vozes (400 Hz a 4 kHz) em telefonia, por exemplo. Tais características definem as restrições sob as quais implementações da tecnologia de dados sobre som têm que operar para serem capazes de empregar microfones e alto-falantes convencionais. Dessa forma, a largura de banda disponível para comunicação DoS restringe-se a um limite máximo em torno de 20 kHz.

A utilização de frequências ultrassônicas é especialmente afetada, permanecendo restrita a frequências entre 15 kHz e 20 kHz em virtualmente qualquer aplicação que não envolva hardware especializado projetado para exceder os limites da audição humana. Sistemas que envolvem algum tipo de compressão de áudio também são inadequados para ultrassom, pois o processo envolve remover frequências ultrassônicas para reduzir o número de bits do sinal.



PROTOCOLOS DOS DE CAMADA FÍSICA

Este capítulo apresenta os protocolos de comunicação DoS estudados em pesquisa para este trabalho.

3.1 Quiet

Quiet é uma biblioteca que utiliza ferramentas da solução código aberto de processamento digital de sinal liquid-dsp para possibilitar transmissão de dados através de som ([QUIET, 2022](#)).

Diversos perfis compostos por conjuntos de parâmetros para configuração da codificação/decodificação de dados são disponibilizados. Existem opções para envio de dados via cabos, que utilizam quase todo o espectro de frequências de áudio disponível para maximizar a taxa de dados, e também opções ultrassônicas codificando dados em frequências acima de 16 kHz visadas para transmissão utilizando alto-falantes, cujos sons não perturbam humanos ao custo de taxas de dados consideravelmente menores, dado o espectro limitado de altas frequências disponível com equipamentos convencionais.

Quiet, que é construída em linguagem C, possui um *binding* JavaScript chamado Quiet.js ([QUIET-JS, 2022](#)) e sua atualização mais recente ocorreu em novembro de 2019.

3.2 AstroMech

AstroMech é um protocolo de comunicação DoS construído em linguagem C++ para a plataforma Arduino ([WECKBACH, 2022](#)). Sua atualização mais recente ocorreu em março de 2019.

O protocolo, capaz de transmitir pequenas quantidades de dados utilizando ondas sonoras, oferece comunicação em um sentido (simplex), com um dispositivo transmitindo um sinal a ser interpretado por um ou mais dispositivos receptores.

O mecanismo de modulação empregado em AstroMech funciona com 16 frequências audíveis (entre 600 Hz e 2850 Hz, com 150 Hz de separação entre símbolos para evitar sobreposição), de forma que cada som emitido pelo transmissor corresponde a quatro bits dos dados codificados. O protocolo opera de forma assíncrona, com transmissor e receptor(es) definindo uma taxa para envio que determinará quantos sons serão emitidos por segundo pelo dispositivo transmissor. Um Código Reed-Solomon é acrescentado ao conteúdo de mensagens pelo protocolo como método de correção de erro para aumento da confiabilidade.

3.3 SoniTALK

SoniTalk é um projeto visando o desenvolvimento de um protocolo aberto e transparente para comunicação DoS com sons na faixa ultrassônica entre dispositivos ([ZEPPELZAUER; RINGOT, 2019](#)). SoniTALK é construído em linguagem Java e sua atualização mais recente ocorreu em julho de 2019.

A privacidade do usuário é uma prioridade do projeto, com um sistema multinível de permissões disponível para oferecer controle ao usuário. O sistema permite que o usuário saiba a todo momento que dados estão sendo transmitidos via SoniTALK e disponibiliza três níveis de privacidade definindo o prazo de vencimento das autorizações concedidas.

O protocolo desenvolvido no projeto transmite dados na forma de mensagens compostas por múltiplos blocos. Os blocos inicial e final de mensagens são utilizados para detecção de uma mensagem contendo, respectivamente, as metades inferior e superior de frequências em ordem ascendente.

3.4 gwave

O gwave é uma biblioteca de dados sobre som construída em linguagem C++ que implementa um protocolo de transmissão baseado em *frequency-shift keying* (FSK) passível de integração em projetos diversos ([GERGANOV, 2022](#)). A taxa de dados varia de 8 a 16 bytes/segundo de acordo com os parâmetros do protocolo, existem variantes audíveis ou ultrassônicas, e códigos de correção de erro (códigos Reed-Solomon, especificamente) são utilizados para aumentar a robustez da transmissão em condições adversas.

Devido a características atrativas apresentadas por gwave em relação aos demais protocolos DoS que analisamos, incluindo estar em desenvolvimento ativo pelo seu criador até o momento, oferecimento de *bindings* Python e JavaScript, possibilidade de comunicação duplex e presença de uma solução para correção de erro, nós o escolhemos como solução da camada física sobre a qual nosso protocolo DoS, de atuação na camada de enlace, é construído. Os capítulos seguintes deste trabalho descrevem detalhes técnicos do gwave e os experimentos que realizamos para avaliar seu desempenho.



PROTOCOLO GGWAVE PARA DADOS SOBRE SOM

O ggwave é uma biblioteca de dados sobre som, cujo protocolo de transmissão utiliza de modulação por chaveamento de frequência (*frequency-shift keying*; FSK).

A taxa de largura de banda das transmissões pode variar de 2 a 16 bytes/s de acordo com os parâmetros escolhidos para o protocolo de transmissão, o qual pode apresentar variações em relação ao número de repetições de cada elemento em uma transmissão e em relação a faixa de frequências empregada (uma faixa de 4,5 kHz é usada e variantes de menor taxa de transferência restritas a 1/3 ou 1/6 de tal faixa – denominadas Dual-Tone e Mono-Tone, respectivamente – são oferecidas para situações em que frequências utilizáveis são limitadas).

A porção do espectro de frequências sonoras empregada nas transmissões pode ser configurada, permitindo codificação de dados na forma de ondas sonoras audíveis ou ondas ultrassônicas de acordo com a aplicação almejada.

O protocolo implementa correção de erro por meio de códigos Reed-Solomon ([GERGA-NOV, 2022](#)), o que permite aumento da robustez da decodificação em situações adversas, como um meio de comunicação que apresenta interferência.

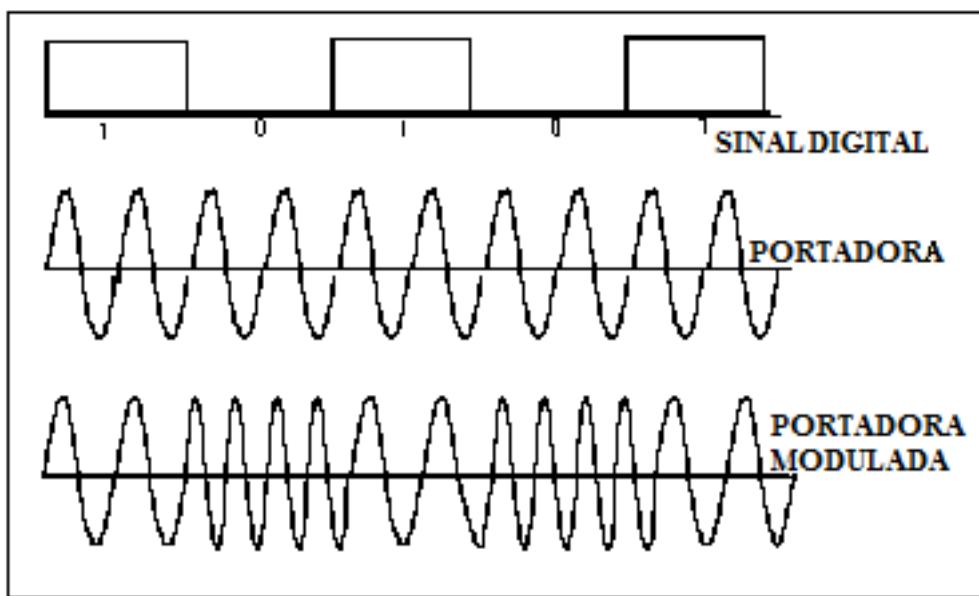
4.1 Transmissão e Camada Física

O algoritmo de modulação/demodulação usado para codificação de dados em ondas sonoras emprega um tipo de modulação de frequência conhecido como FSK. Tal esquema envolve manipulação da onda portadora¹ na forma de variações de sua frequência entre frequências individuais diversas para transmissão de informação digital. Em sua variante mais simples,

¹ Onda que será modificada com base em um sinal para se tornar uma representação da informação nele contida.

FSK faz uso de duas frequências individuais para transmissão dos valores binários “0” e “1”. Isso é exemplificado na Figura 1, em que uma onda portadora é apresentada sem modulação inicialmente e em seguida já modulada de acordo com o sinal digital (exibindo uma frequência mais baixa representando o estado “1” e uma frequência mais alta representando o estado “0”). A biblioteca ggwave emprega FSK com a presença de mais valores de frequência, o que possibilita a codificação de mais bits por valor de frequência em relação ao único bit permitido por duas frequências.

Figura 1 – Duas frequências diferentes são utilizadas para representar o estado “1” e o estado “0” de um sinal digital na variante binária da modulação FSK.



Fonte: Adaptada de [Cadete \(2010\)](#).

Os dados a serem transmitidos são divididos em porções de quatro bits. Durante a transmissão, o protocolo utiliza uma faixa de frequências sonoras de até 4,5 kHz, dividida em frequências individuais espaçadas com saltos iguais de 46,875 Hz entre cada uma (a frequência inicial e final de tal faixa varia de acordo com parâmetros do protocolo escolhidos). Até três bytes dos dados podem ser transmitidos a cada momento, com uma forma de onda composta por soma de até seis sons dentre as frequências da faixa sonora sendo utilizada. Um som nesse contexto consiste de uma onda sonora sinusoidal² composta por uma única frequência, sendo que qualquer onda mais complexa pode ser decomposta em um certo conjunto de tais ondas através de uma técnica matemática conhecida como transformada de Fourier. O valor de frequência de cada som combinado para produção da forma de onda a ser transmitida codifica quatro bits, correspondendo a uma das porções em que os dados são divididos.

O diagrama apresentado na Figura 2 contém uma representação simplificada da trans-

² Oscilação periódica simples descrita por uma função seno ou cosseno.

missão de uma mensagem. A faixa de frequências de 4,5 kHz é apresentada evidenciando as frequências individuais igualmente espaçadas, totalizando 96 frequências, utilizadas na modulação FSK empregada por gwave. Cada conjunto de 16 frequências consecutivas foi evidenciado, indicando que consistem nos seis segmentos dos quais as frequências dos sons que compõem cada forma de onda se originam. A frequência inicial de um desses segmentos representa no esquema de modulação o valor binário 0000 e a frequência final o valor binário 1111, possibilitando que o valor de frequência de cada som codifique quatro bits de informação. A forma de onda gerada por gwave para transmissão é composta por seis sons, cada um derivado de um dos seis segmentos indicados da faixa de frequências, de modo que três bytes da informação a ser transmitida podem ser representados por ela a cada momento. No diagrama, o texto "Alfa", codificado em Unicode UTF-8, é utilizado para exemplificar o processo de codificação de informação. Os dados a serem transmitidos são divididos em porções de quatro bits, identificadas por P0 - P7 no diagrama, que são representadas por um som proveniente de cada segmento da faixa de frequência como demonstrado.

A biblioteca gwave possui disponíveis protocolos para transmissões audíveis e transmissões ultrassônicas. Ambos empregam uma faixa de frequências sonoras de 4,5 kHz, tendo como frequências iniciais 1875 Hz para os protocolos audíveis e 15000 Hz para os protocolos ultrassônicos. Protocolos empregando dois sons ou um som ao invés de seis na codificação dos dados em ondas sonoras estão disponíveis — com uma redução da taxa de transferência de dados possível em proporção equivalente — permitindo utilização de uma faixa de frequências menor que 4,5 kHz. Os protocolos utilizando dois sons, chamados de dual-tone, empregam uma faixa sonora de 1,5 kHz e os protocolos utilizando um som, chamados mono-tone, uma faixa de 750 Hz, com a frequência inicial para ambos de 1125 Hz.

Os protocolos dual-tone e mono-tone possibilitam transmissão de dados em situações que apresentam limitações em relação a faixa de frequências disponível para uso. Certos dispositivos de áudio têm capacidades modestas em relação a faixa de frequências que são capazes de reproduzir adequadamente, de modo que os 4,5 kHz requeridos pelos demais protocolos gwave inviabilizariam o emprego de tais dispositivos em um projeto.

O diagrama apresentado na Figura 3 contém uma representação simplificada da transmissão de uma mensagem utilizando protocolos dual-tone. Diferentemente dos protocolos que empregam a faixa de frequências de 4,5 kHz, as variantes dual-tone se restringem a apenas 1,5 kHz, totalizando 32 frequências individuais igualmente espaçadas, no processo de modulação. Sendo assim, a forma de onda gerada por gwave para transmissão é composta por dois sons, cada um derivado de um dos dois segmentos indicados da faixa de frequências, de modo que cada som é capaz de transmitir 1 byte de informação. O texto "Alfa", codificado em Unicode UTF-8, é utilizado para exemplificar o processo de codificação de informação desses protocolos no diagrama. Os dados a serem transmitidos são divididos em porções de quatro bits, identificadas por P0 - P7, que são representadas por um som proveniente de cada segmento da faixa de

Figura 2 – Exemplo de codificação em ggwave de "Alfa", correspondente a: [0100, 0001]; [0110, 1100]; [0110, 0110]; [0110, 0001].

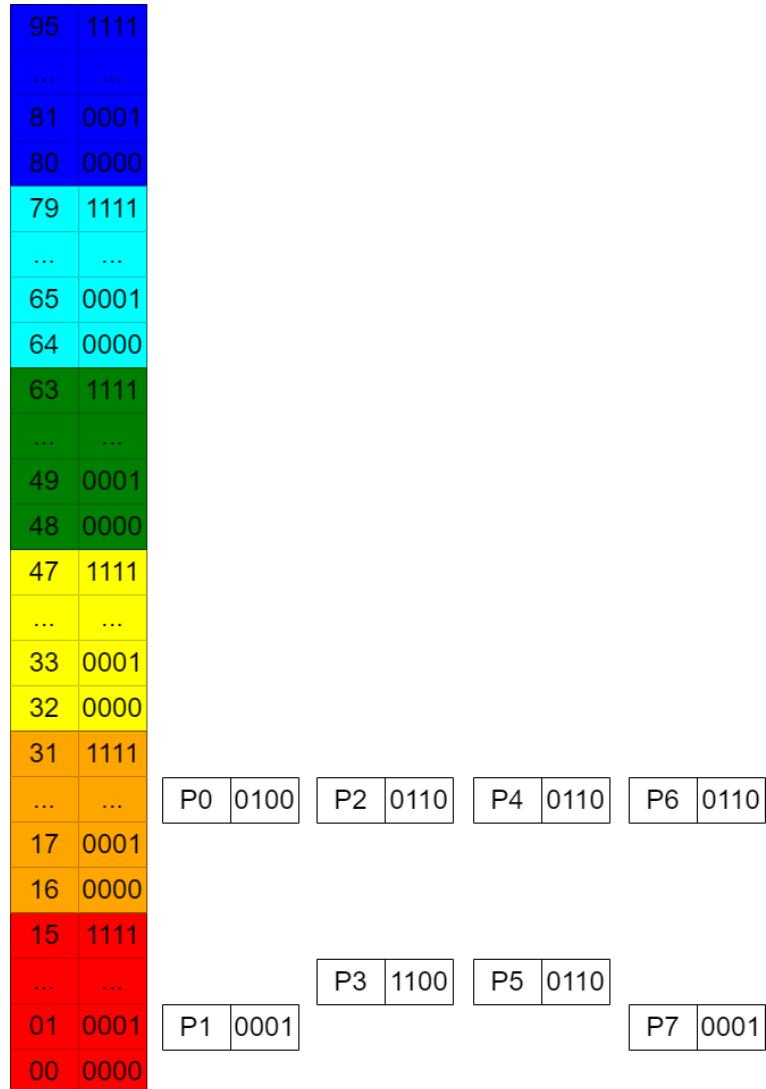
| | | | |
|-----|------|----|------|
| 95 | 1111 | | |
| ... | ... | P4 | 0110 |
| 81 | 0001 | | |
| 80 | 0000 | | |
| 79 | 1111 | | |
| ... | ... | P5 | 0110 |
| 65 | 0001 | | |
| 64 | 0000 | | |
| 63 | 1111 | | |
| ... | ... | P2 | 0110 |
| 49 | 0001 | | |
| 48 | 0000 | | |
| 47 | 1111 | | |
| ... | ... | P3 | 1100 |
| 33 | 0001 | | |
| 32 | 0000 | | |
| 31 | 1111 | | |
| ... | ... | P0 | 0100 |
| 17 | 0001 | | |
| 16 | 0000 | | |
| 15 | 1111 | | |
| ... | ... | P6 | 0110 |
| 01 | 0001 | | |
| 00 | 0000 | P1 | 0001 |
| | | | P7 |
| | | | 0001 |

Fonte: Elaborada pelo autor.

frequência como demonstrado.

Os protocolos ggwave têm como parâmetros de funcionamento sua frequência inicial, quantidade de transmissões por cada conjunto de bytes e o número de bytes transmitidos a cada momento. A frequência inicial especifica em que ponto do espectro de áudio começa a faixa de frequências a ser utilizada para transmissão. A quantidade de transmissões por conjunto de bytes determina a taxa de dados possível, de forma que valores menores oferecem transmissões com taxas de dados maiores ao custo de potencial redução da robustez de sua recepção e vice-versa. O número de bytes transmitidos a cada momento define se seis sons são empregados em transmissões, fazendo uso da totalidade de uma faixa de frequências de 4,5 kHz, ou uma quantidade reduzida para uso de faixa de frequências mais restrita (protocolos dual-tone e mono-tone). A Figura 4 apresenta os parâmetros de funcionamento de protocolos contidos no arquivo

Figura 3 – Exemplo de codificação dual-tone em ggwave de "Alfa", correspondente a: [0100, 0001]; [0110, 1100]; [0110, 0110]; [0110, 0001].



Fonte: Elaborada pelo autor.

header de ggwave.

Três taxas de transferência de dados são oferecidas para cada tipo de protocolo. Tal configuração altera o número de repetições da cada som durante transmissões, tendo impacto sobre a precisão da recepção de mensagens. Para os protocolos audíveis e ultrassônicos, a configuração *Normal* se traduz em uma taxa de transferência de 11,17 Bps, a configuração *Fast* em uma taxa de 16,76 Bps e *Fastest* em 32,52 Bps. Os protocolos dual-tone têm taxas de transferência correspondentes a 1/3 dos valores citados e, no caso dos protocolos mono-tone, 1/6 dos valores citados. A Figura 5 apresenta as configurações de protocolos contidas no arquivo *header* de ggwave.

Qualquer instância ggwave criada tentará decodificar transmissões recebidas utilizando os protocolos de recepção ativados e será capaz de transmitir dados utilizando os protocolos de

Figura 4 – Parâmetros de funcionamento de protocolos ggwave.

```

struct Protocol {
    const char * name; // string identifier of the protocol

    int16_t freqStart; // FFT bin index of the lowest frequency
    int8_t framesPerTx; // number of frames to transmit a single chunk of data
    int8_t bytesPerTx; // number of bytes in a chunk of data
    int8_t extra; // 2 if this is a mono-tone protocol, 1 otherwise

    bool enabled;

    int nTones() const { return (2*bytesPerTx)/extra; }
    int nDataBitsPerTx() const { return 8*bytesPerTx; }
    int txDuration_ms(int samplesPerFrame, float sampleRate) const {
        return framesPerTx*((1000.0f*samplesPerFrame)/sampleRate);
    }
};


```

Fonte: Elaborada pelo autor.

Figura 5 – Configurações de protocolos oferecidas por ggwave.

```

protocols.data[GGWAVE_PROTOCOL_AUDIBLE_NORMAL] = { GGWAVE_PSTR("Normal"), 40, 9, 3, 1, true, };
protocols.data[GGWAVE_PROTOCOL_AUDIBLE_FAST] = { GGWAVE_PSTR("Fast"), 40, 6, 3, 1, true, };
protocols.data[GGWAVE_PROTOCOL_AUDIBLE_FASTEST] = { GGWAVE_PSTR("Fastest"), 40, 3, 3, 1, true, };
protocols.data[GGWAVE_PROTOCOL_ULTRASOUND_NORMAL] = { GGWAVE_PSTR("[U] Normal"), 320, 9, 3, 1, true, };
protocols.data[GGWAVE_PROTOCOL_ULTRASOUND_FAST] = { GGWAVE_PSTR("[U] Fast"), 320, 6, 3, 1, true, };
protocols.data[GGWAVE_PROTOCOL_ULTRASOUND_FASTEST] = { GGWAVE_PSTR("[U] Fastest"), 320, 3, 3, 1, true, };

protocols.data[GGWAVE_PROTOCOL_DT_NORMAL] = { GGWAVE_PSTR("[DT] Normal"), 24, 9, 1, 1, true, };
protocols.data[GGWAVE_PROTOCOL_DT_FAST] = { GGWAVE_PSTR("[DT] Fast"), 24, 6, 1, 1, true, };
protocols.data[GGWAVE_PROTOCOL_DT_FASTEST] = { GGWAVE_PSTR("[DT] Fastest"), 24, 3, 1, 1, true, };
protocols.data[GGWAVE_PROTOCOL_MT_NORMAL] = { GGWAVE_PSTR("[MT] Normal"), 24, 9, 1, 2, true, };
protocols.data[GGWAVE_PROTOCOL_MT_FAST] = { GGWAVE_PSTR("[MT] Fast"), 24, 6, 1, 2, true, };
protocols.data[GGWAVE_PROTOCOL_MT_FASTEST] = { GGWAVE_PSTR("[MT] Fastest"), 24, 3, 1, 2, true, };


```

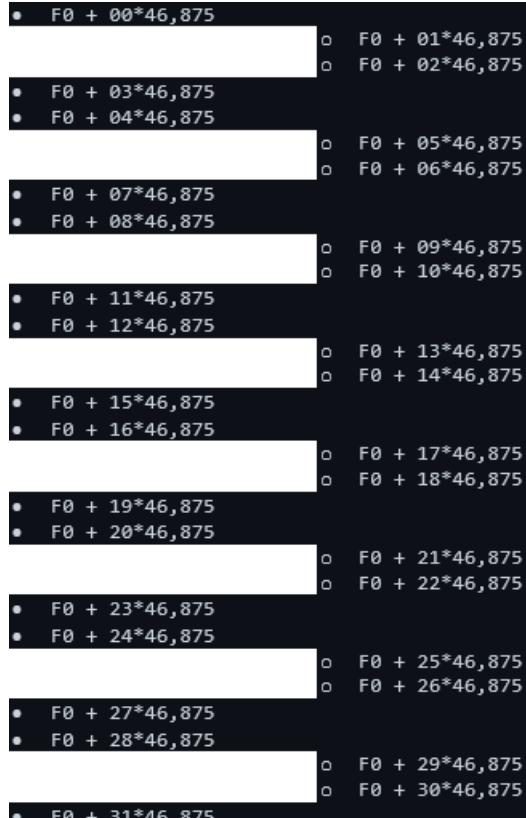
Fonte: Elaborada pelo autor.

transmissão ativados. Ativar somente os protocolos recepção e transmissão necessários otimiza a performance e uso de memória da biblioteca. O processo de decodificação é simplificado quando protocolos não necessários são desativados, o que contribui para aumento da precisão ao decodificar mensagens e redução da ocorrência de falsos positivos.

O início e fim de transmissões é demarcado com marcadores predefinidos para que um receptor seja capaz de identificá-los e então gravar o áudio entre eles para decodificação de frequências. O marcador de início é composto por 16 das 32 frequências iniciais da faixa de transmissão e o marcador de fim é composto pelas 16 frequências restantes. As 32 frequências iniciais de amostras de entrada são analisadas com comparação da força relativa de frequências vizinhas para identificação de um marcador. Para evitar que ruído aleatório seja capaz de produzir um marcador, 16 frequências são empregadas para manter mínimo o número de falsos positivos. A Figura 6 apresenta as disposições de frequências dos marcadores de início e fim, com F0

representando 1875 Hz para protocolos audíveis, 15000 Hz para protocolos ultrassônicos e 1125 Hz para protocolos dual-tone.

Figura 6 – Disposição de frequências do marcador de início e do marcador de fim.



Fonte: Elaborada pelo autor.

Considerando um número x de bytes de dados a serem transmitidos, o número de bytes que são de fato transmitidos corresponde a $3 + x + \text{comprimentoECC}(x)$. Três bytes são reservados para especificação do comprimento da mensagem (1 byte + 2 para ECC). A função $\text{comprimentoECC}(x)$ consiste no número de bytes de correção de erro utilizados para comprimento x , determinando que, se o comprimento é menor que 4, retorna-se 2; se o comprimento é maior ou igual 4, retorna-se $\max(4, 2^*(\text{comprimento}/5))$.

A biblioteca ggwave utiliza para transformada rápida de Fourier (FFT) uma implementação preexistente ([OOURA, 2006](#)).

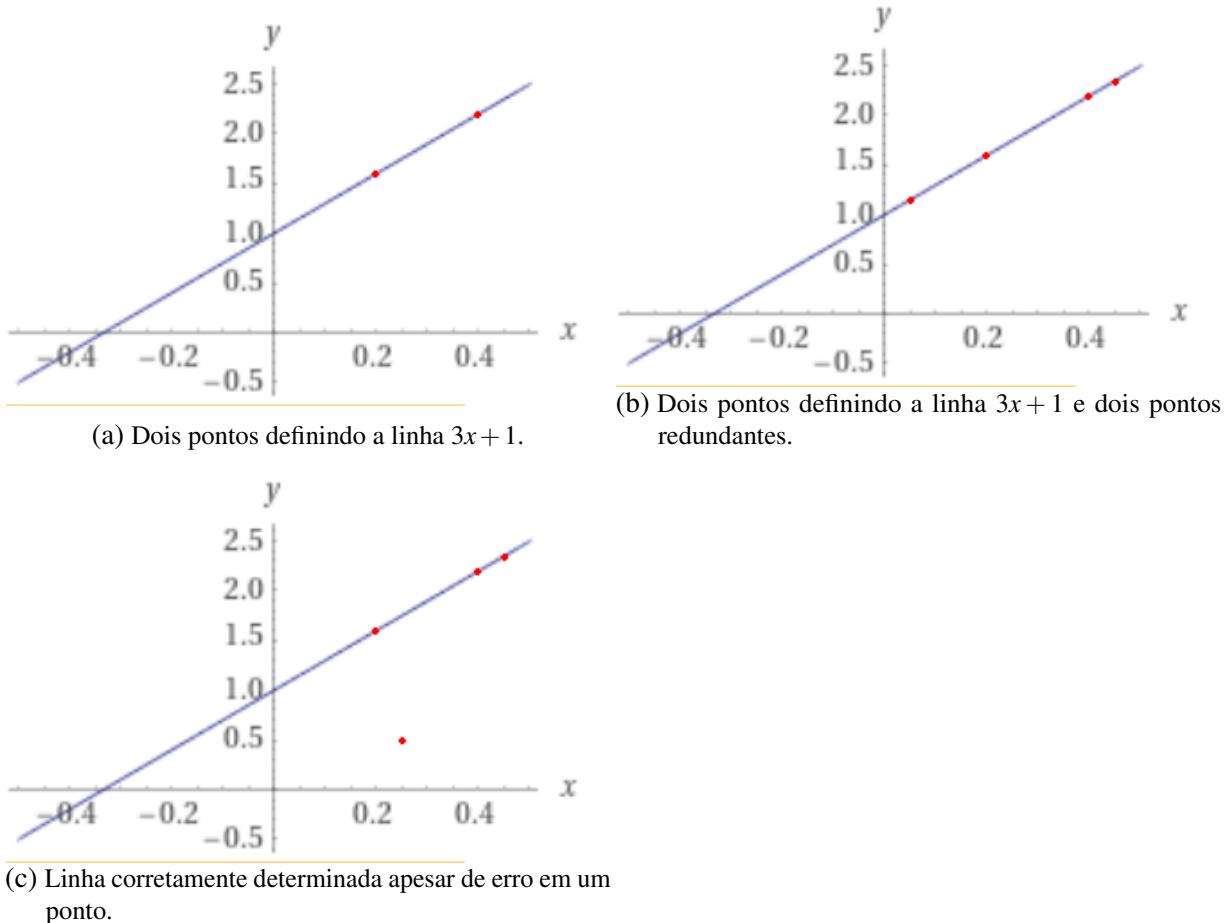
4.2 Correção de erros

Os dados de uma mensagem são transmitidos acrescidos de códigos de correção de erro Reed-Solomon (ECC) ([GERGANOV, 2022](#)). O uso de Reed-Solomon permite a identificação e correção de bytes corrompidos de uma mensagem se a quantidade de erros não ultrapassa metade do número de bytes ECC contidos na mensagem. A quantidade de bytes ECC em uma mensagem depende do tamanho dos dados que a constituem.

Códigos de correção de erro baseiam-se na adição de informação redundante aos dados a ser transmitidos por um canal de comunicação ruidoso para permitir que o receptor seja capaz de deduzi-los corretamente caso erros sejam introduzidos ao longo do processo. Códigos de correção de erro como a variante Reed-Solomon em questão calculam bytes ECC como função dos bytes que compõem a mensagem, de modo que cada mensagem possível de ser gerada tem um conjunto específico de bytes ECC correspondentes, e enviam os bytes da mensagem diretamente em conjunto com bytes ECC acrescidos ao final.

Os códigos Reed-Solomon se baseiam na propriedade de que todo polinômio de grau n é unicamente determinado por $n + 1$ pontos. Por exemplo, um polinômio de grau um como $3x + 1$ é determinado por dois pontos: em uma situação hipotética na qual deseja-se enviar dois pontos definindo a linha resultante por um canal de comunicação, caso os pontos fossem acompanhados de dois pontos adicionais da mesma linha, seria possível corrigir um erro de transmissão encaixando uma linha — que resultaria na exclusão do ponto errôneo — entre os pontos recebidos. A Figura 7 apresenta uma visualização de tal exemplo.

Figura 7 – Polinômio exemplificando princípio por trás de códigos Reed-Solomon.



Fonte: Elaborada pelo autor.

Os Códigos Reed-Solomon são definidos como polinômios que operam sobre estruturas

algebricas conhecidas como corpos de Galois (REED; SOLOMON, 1960), que são conjuntos finitos de números em que operações aritméticas sobre seus elementos resultam sempre em um elemento que pertence ao conjunto. O processo de codificação envolve divisão do polinômio que representa a mensagem por um polinômio gerador que é derivado de acordo com o número de símbolos de correção de erro escolhidos, com o resto dessa divisão correspondendo ao código Reed-Solomon que é acrescido ao fim da mensagem original. O processo de decodificação envolve determinar se as raízes do polinômio gerador correspondem às raízes do polinômio que representa a mensagem. O caso de correspondência do resultado indica que a mensagem não contém nenhum erro. Em caso contrário, os resultados dessas operações contêm a informação necessária para determinação das correções a serem feitas. A detecção durante decodificação de quais caracteres de uma mensagem transmitida foram corrompidos é realizada com um algoritmo como Berlekamp-Massey para identificação eficiente do polinômio localizador de erros, dada a impossibilidade de encontrar a palavra de código permitida mais próxima em uma lista completa por métodos de busca exaustiva considerando o número excessivo de possibilidades.

Para símbolos de m bits, as palavras de código (unidade contendo dados e ECC) têm comprimento de $2^m - 1$ símbolos. Símbolos de 8 bits (cada símbolo tem 256 valores possíveis, correspondendo a um byte) são comumente utilizados, levando a palavras de código de 255 bytes ($2^8 - 1$). Uma técnica para encurtamento da uma palavra de código é utilizada caso os dados a ser transmitidos sejam insuficientes para preencher a completamente: a porção não utilizada é complementada pela inserção de zeros antes da codificação, a transmissão da palavra de código gerada ocorre excluindo os zeros inseridos, a decodificação da palavra de código é realizada com reinserção dos zeros previamente removidos.

O número de bytes ECC acrescidos a uma mensagem corresponde a 40% do comprimento de dados a ser transmitidos. Dado um número x de bytes a ser transmitidos, a função $\text{ComprimentoECC}(x) : x < 4?2 : \max(4, 2 * (x/5))$ é empregada para definir o número de bytes ECC utilizados. A implementação de correção de erros é capaz de codificar até 256 caracteres por mensagem devido ao emprego de um corpo de Galois composto por um conjunto de 256 elementos (corpo de Galois 2^8). Um corpo de Galois com mais elementos leva a grandes polinômios, tornando custosas (tempo quadrático) as computações envolvidas no processo de codificação/decodificação. Considerando que 40% do tamanho de mensagens é dedicado para bytes ECC, cada transmissão limita-se a até 183 bytes de conteúdo ($183 + 40\% \text{ ECC} = 256$).

A biblioteca ggwave utiliza como solução para correção de erros uma implementação preexistente de codificador/decodificador de códigos Reed-Solomon disponível no GitHub ([LUBINETS, 2022](#)).

4.3 Exemplo

Um exemplo de código de uso em uma situação básica de transmissão (Código-fonte 1) e recepção (Código-fonte 2) de uma mensagem com texto fazendo uso dos *bindings* Python de ggwave disponíveis.

Código-fonte 1 – Código Python para codificação e transmissão de uma mensagem com texto (texto exemplo: "Alfa") por som utilizando ggwave

```

1: import ggwave
2: import pyaudio
3:
4: p = pyaudio.PyAudio()
5:
6: # geração de forma de onda de audio para string "Alfa"
7: waveform = ggwave.encode("Alfa", protocolId = 1, volume = 20)
8:
9: print("Transmitindo 'Alfa'...")
10: stream = p.open(format=pyaudio.paFloat32, channels=1, rate
11:                  =48000, output=True, frames_per_buffer=4096)
12: stream.write(waveform, len(waveform)//4)
13: stream.stop_stream()
14: stream.close()
15: p.terminate()
```

A função `encode()` em ggwave é responsável pela codificação de um *payload* em uma forma de onda de áudio. Ela recebe como parâmetro uma *string* contendo o *payload* e retorna os bytes da forma de onda de áudio gerada representando amostras inteiras com sinal de 16 bits.

Código-fonte 2 – Código Python para captura e decodificação de informação em áudio utilizando ggwave

```

1: import ggwave
2: import pyaudio
3:
4: p = pyaudio.PyAudio()
5:
6: stream = p.open(format=pyaudio.paFloat32, channels=1, rate
6:                  =48000, input=True, frames_per_buffer=1024)
7:
8: print('Ouvindo... Use Ctrl+C para parar')
9: instance = ggwave.init()
```

```
10:  
11: try:  
12:     while True:  
13:         data = stream.read(1024, exception_on_overflow=False)  
14:         res = ggwave.decode(instance, data)  
15:         if (not res is None):  
16:             try:  
17:                 print('Transmissao recebida: ' + res.decode()  
'utf-8'))  
18:             except:  
19:                 pass  
20: except KeyboardInterrupt:  
21:     pass  
22:  
23: ggwave.free(instance)  
24:  
25: stream.stop_stream()  
26: stream.close()  
27:  
28: p.terminate()
```

A função `decode()` em `ggwave` é responsável pela análise e decodificação de uma forma de onda de áudio objetivando obtenção do *payload* original. Ela recebe como parâmetro os bytes da forma de onda de áudio a se decodificar e, se bem sucedida, retorna o *payload* decodificado.



EXPERIMENTOS COM GGWAVE

Este capítulo apresenta experimentos para avaliação do desempenho do protocolo de dados sobre som do ggwave. Os experimentos avaliaram o efeito da distância, de ruído ambiente e obstáculos físicos na taxa de sucesso da transmissão de mensagens. Nos testes conduzidos, nós contabilizamos o número de mensagens recebidas corretamente por um dispositivo receptor, em diferentes distâncias e de acordo com a presença ou ausência de obstrução física, em uma sala fechada silenciosa ou com ruído ambiente.

5.1 Ambiente de Experimentação

Os experimentos foram baseados na versão 0.4.2 do ggwave através de seus *bindings* Python, em computadores com Python 3.10.4 e a biblioteca PyAudio 0.2.12¹. Nós utilizamos um código nos testes para a transmissão a cada três segundos de uma mensagem.

Os testes avaliaram o desempenho das transmissões audíveis e ultrassônicas de ggwave em seus protocolos padrões (`protocolId = 1` e `protocolId = 4`, respectivamente). Ambas as variantes empregam uma faixa de frequências sonoras de 4,5 kHz, tendo como frequência inicial 1875 Hz para o protocolo audível e 15000 Hz para o protocolo ultrassônico. A taxa de transferência de dados utilizada nesses protocolos é a configuração *Fast*, que corresponde a uma taxa de 16,76 B/s.

Nós conduzimos o experimento no Laboratório de Sistemas Digitais da Universidade Federal de Catalão (Sala 16 do Bloco J). Os dispositivos utilizados consistem em um microfone de mesa e uma caixa de som externa conectados a computadores por cabos P2 estéreo de 3,5 mm, conforme a Figura 8. A Figura 9 apresenta as especificações técnicas dos dispositivos, assim como dos computadores.

Para os testes, nós ajustamos a sensibilidade do microfone de mesa de acordo com os

¹ <<https://pypi.org/project/PyAudio/0.2.12/>>

Figura 8 – Microfone de mesa e caixa de som externa utilizados para o experimento.



(a) Transmissor.



(b) Receptor.

Fonte: Elaborada pelo autor.

Figura 9 – Especificações técnicas dos dispositivos utilizados para o experimento.

| Microfone | |
|-------------------------|-----------------|
| Redução de ruído | Não |
| Resposta em frequência | 100 – 12.000 Hz |
| Impedância | 2.200 ohms |
| Sensibilidade | -45 dB |
| Nível de pressão de som | 115 dB |
| Padrão de captação | Omnidirecional |
| Relação sinal-ruído | 58 dB |
| Tipo de sensor | Condensador |

| Caixa de som | |
|-------------------------|------------------------|
| Potência de saída | 3 watts |
| Impedância | 4 ohms |
| Intervalo de frequência | 280 Hz – 16.000 Hz |
| Relação sinal-ruído | Maior ou igual a 95 dB |

| Computador receptor | |
|----------------------------|-----------------------------|
| Processador | Intel Core i5-2400 3,10 GHz |
| Sistema operacional | Lubuntu 22.04 LTS |
| Interface de memória | DDR3 SDRAM |
| Velocidade de disco | 7200 RPM |
| Disco rígido | 500 GB |
| Interface de disco | SATA |
| Memória | 8 GB |

| Computador transmissor | |
|-------------------------------|------------------------------|
| Processador | Intel Core i3-3217U 1,80 GHz |
| Sistema operacional | Linux Mint 20.1 MATE 64-bit |
| Interface de memória | DDR3 SDRAM |
| Velocidade de disco | 5400 RPM |
| Disco rígido | 500 GB |
| Interface de disco | SATA |
| Memória | 2 GB |

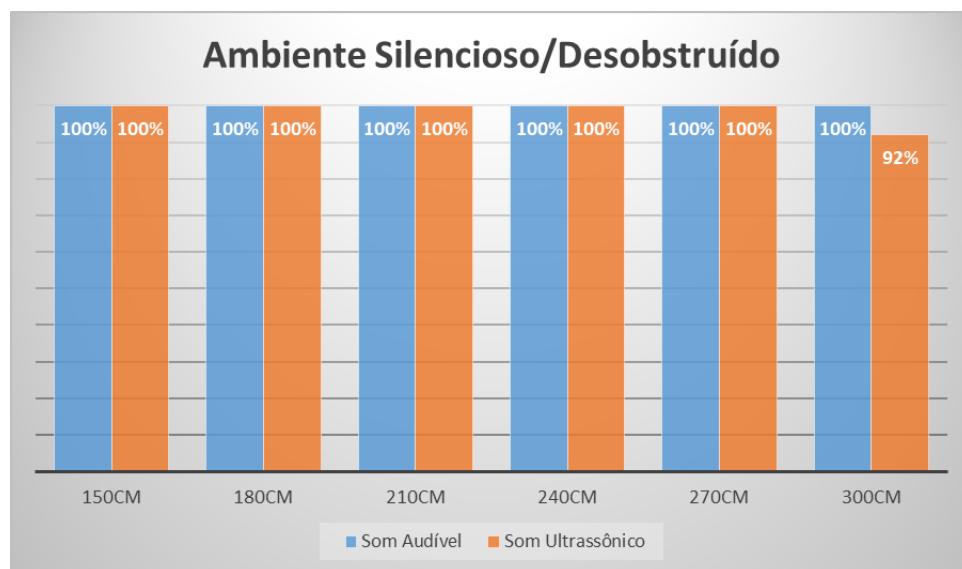
Fonte: Elaborada pelo autor.

níveis de som captados por ele no cenário silencioso do ambiente de experimentação, de modo a mantê-los todos abaixo de aproximadamente -30 dB, confortavelmente distantes do nível máximo que ele é capaz de reproduzir. Com esse objetivo, aplicamos uma atenuação de -18,06 dB ao microfone, o que correspondeu à configuração de 50% do volume da entrada de áudio do computador conectado. Configuramos o volume de 100% para a saída de áudio do computador conectado à caixa de som externa.

5.2 Efeito de Distância na Transmissão

Para avaliar a influência de distância na transmissão, nós variamos a distância entre transmissor e receptor, variando de 150 a 300 cm em incrementos de 30 cm após cada avaliação. O experimento consistiu no envio de 25 mensagens de 60 bytes de comprimento em intervalos de três segundos. Ao final, coletamos o número de mensagens recebidas com sucesso em cada situação analisada. A Figura 10 apresenta os resultados de percentual de sucesso de recepção de mensagens de acordo com distância.

Figura 10 – Taxa de sucesso na entrega de mensagens em relação à distância no cenário silencioso e sem obstruções.



Fonte: Elaborada pelo autor.

Ambos os protocolos testados atingiram desempenho máximo nesse cenário ao longo de todas as distâncias avaliadas, excetuando a perda de duas mensagens transmitidas a 300 centímetros do receptor com o protocolo ultrassônico. Nós especulamos que tal resultado levemente inferior verificado em transmissões ultrassônicas pode ser atribuído ao fato de que a faixa de frequências em que operam não corresponde aos intervalos de resposta uniforme dos dispositivos utilizados para emissão e captação de áudio. As especificações técnicas desses dispositivos, listadas na Figura 9, indicam que a faixa de frequências acima de 15000 Hz do protocolo ultrassônico está além das frequências que a caixa de som e especialmente o microfone são projetados para reproduzir idealmente.

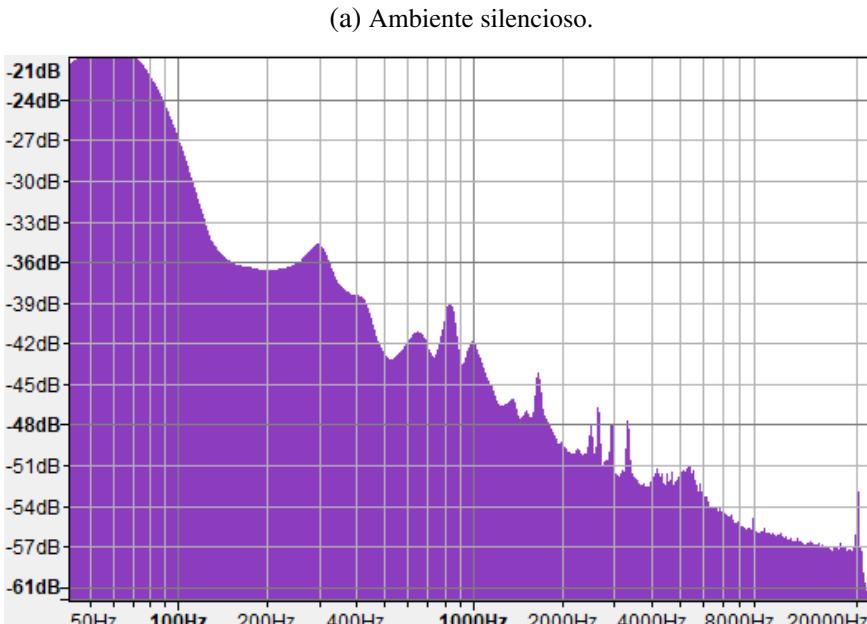
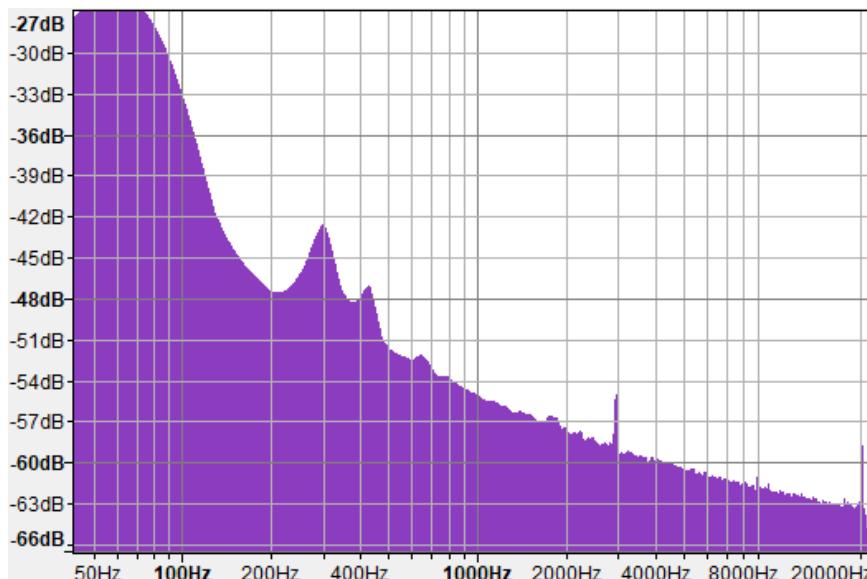
A partir dos testes realizados inferimos que, sem a presença de obstáculos físicos ou ruído ambiente significativo, transmissões com taxa de sucesso total de recepção de mensagens entre dispositivos separados por menos de três metros são possíveis. Entretanto, caso a distância entre os componentes seja de pelo menos três metros, consideração adicional é necessária para assegurar que os equipamentos de áudio empregados apresentem desempenho satisfatório em relação a faixa de frequências selecionada para transmissão.

5.3 Influência de Ruídos no Ambiente

Para avaliar a influência de ruídos na transmissão, aproveitamos ruídos encontrados no próprio laboratório, produzidos por duas fontes: um ar-condicionado e um switch de rede em operação. O ruído gerado pelo switch quando ligado é anômalo e de alta intensidade.

A Figura 11 apresenta o espectro de frequência do som captado com os aparelhos ligados e desligados. O gráfico foi gerado com auxílio do editor de áudio Audacity² no laboratório com e sem as fontes de ruído.

Figura 11 – Comparaçao de espectro de frequência de ambiente silencioso e com ruído.

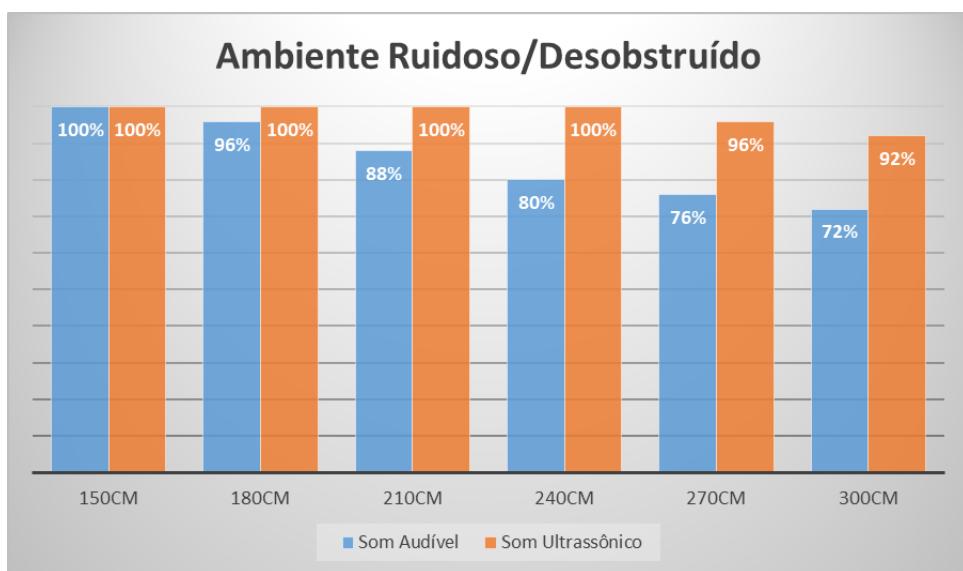


Fonte: Elaborada pelo autor.

² <<https://www.audacityteam.org/>>

O gráfico da Figura 12 apresenta o percentual de sucesso de recepção de mensagens em relação à distância com presença do ruído ambiente caracterizado. Os testes realizados durante o experimento demonstraram o efeito que ruído ambiente pode exercer sobre a taxa de erros verificados dentre as mensagens transmitidas. A presença de ruído ambiente teve grande efeito sobre a qualidade da comunicação quando utilizamos o protocolo audível, especialmente a medida que a distância entre dispositivos aumenta, com perdas de mensagens superiores a 25% na maior distância entre transmissor e receptor. O protocolo ultrassônico, apresentou desempenho essencialmente equivalente aos seus resultados em ambiente silencioso, expostos na Figura 10.

Figura 12 – Taxa de sucesso na entrega de mensagens em relação à distância no cenário ruidoso e sem obstruções.



Fonte: Elaborada pelo autor.

A comparação de espectro de frequência entre ambiente silencioso e com ruído presente na Figura 11 possibilita verificar aumento significativo da intensidade do sinal captado pelo microfone, com elevação geral de pelo menos seis decibéis. A porção de espectro em que o protocolo audível opera (entre 1875 Hz e 6375 Hz), apresenta ruído entre -57 dB e -60 dB quando os aparelhos não estão em operação, valores que se elevam para entre -48 dB e -51 dB quando estão ligados, um aumento expressivo de nove decibéis. Essa elevação significativa do nível de ruído presente levou a degradação de desempenho do protocolo audível verificada em relação aos resultados registrados em ambiente silencioso.

Os espectros de frequência indicam que a intensidade de ruído presente na faixa empregada pelo protocolo ultrassônico (entre 15000 Hz e 19500 Hz) é significativamente inferior ao restante do espectro em ambas as situações. O ambiente silencioso apresenta ruído inferior a -63 dB na faixa quase ultrassônica, e o ambiente com ruído, embora contenha uma elevação da intensidade dessas frequências para valores inferiores -57 dB, concentra suas frequências mais intensas em porções mais baixas do espectro. Nós atribuímos a similaridade do desempenho do protocolo ultrassônico com ou sem a presença de ruído ao fato de que a intensidade de frequê-

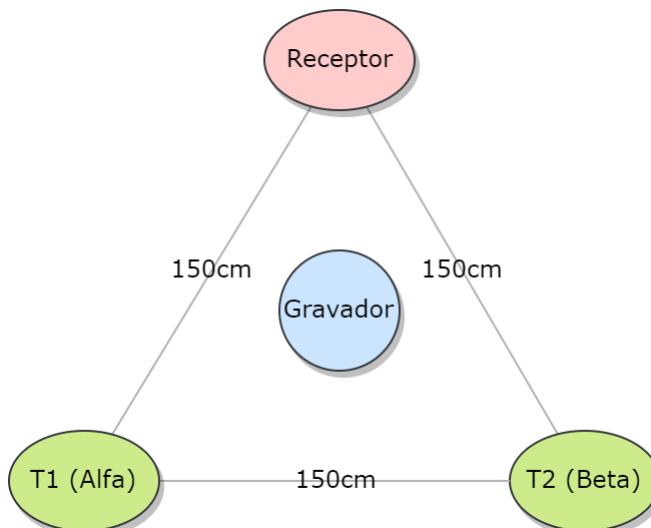
cias em sua faixa de operação permaneceu, apesar das diferenças verificadas, similarmente baixa em ambos os cenários, não ultrapassando -57 dB.

5.4 Interferências entre Transmissões Simultâneas

Em um cenário alternativo, realizamos testes para determinação do efeito de múltiplos dispositivos transmitindo de forma simultânea. A configuração para esses testes envolveu um dispositivo receptor recebendo mensagens diferentes (palavra 'Alfa' e palavra 'Beta') de dois transmissores operando simultaneamente.

Nós capturamos as diversas transmissões nesse cenário, com alternância entre os protocolos oferecidos por gwave, e geramos espectrogramas³ das gravações com o editor de áudio Audacity para análise, visualização e maior compreensão dos resultados coletados. Utilizamos um smartphone (Asus ZenFone 5 T00J com Android 5.0) rodando seu aplicativo padrão de gravação de áudio para as gravações. Devido à compressão com perda de dados nos arquivos gerados pelo aplicativo eliminarem áudio na faixa quase ultrassônica empregada pelos protocolos inaudíveis de gwave, registramos apenas transmissões com protocolo audível para análise com espectrograma. A Figura 13 ilustra a disposição utilizada para os dispositivos no ambiente durante realização dos testes de comunicação simultânea.

Figura 13 – Representação da disposição de elementos utilizada no teste de comunicação simultânea.



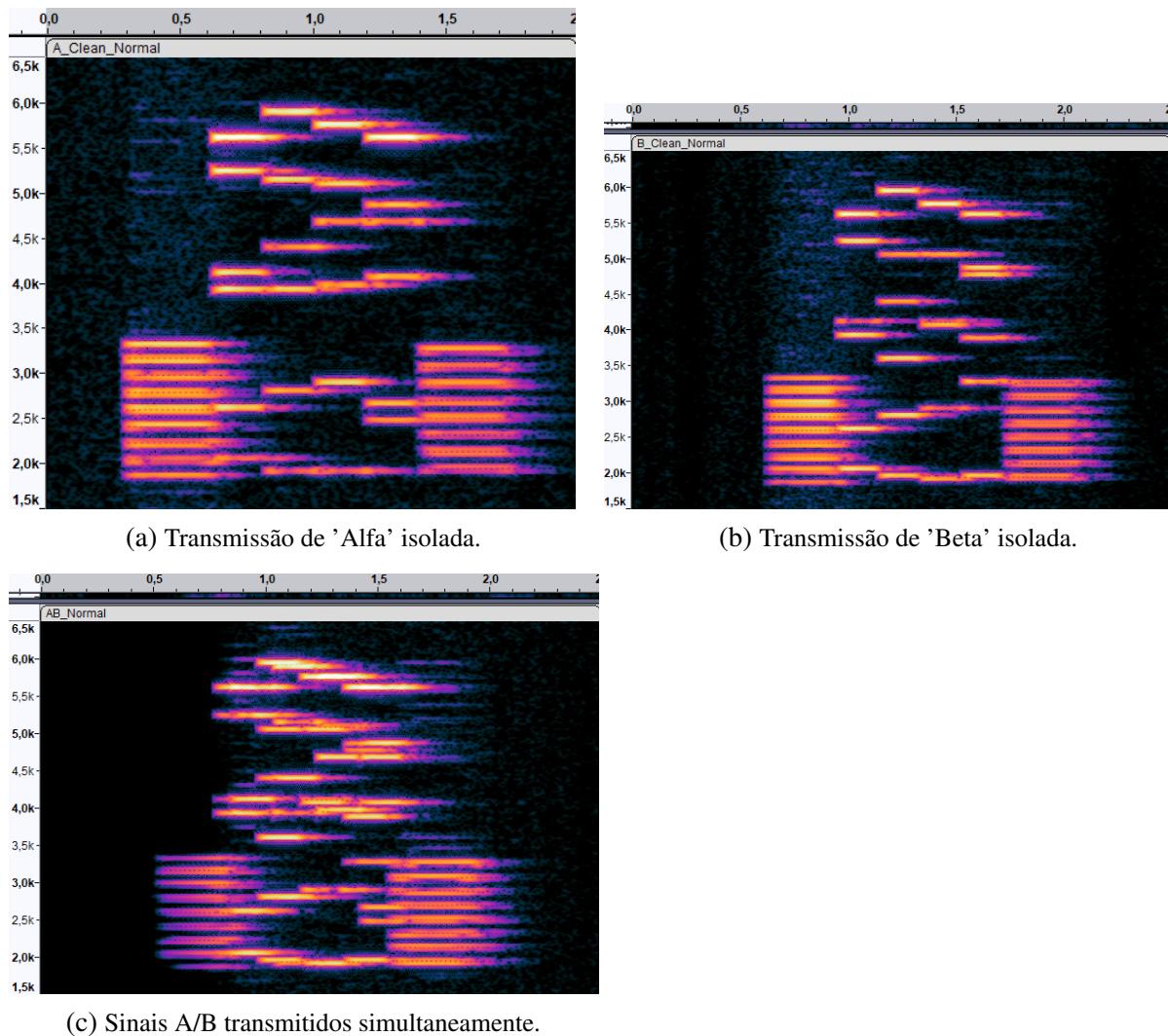
Fonte: Elaborada pelo autor.

A partir dos testes realizados, determinamos que o sinal emitido por dispositivos ao transmitir uma mensagem pode ser severamente afetado por sinais alheios no meio de transmissão. A situação testada no experimento de comunicação simultânea se mostrou suficiente para inviabilizar completamente a comunicação entre dispositivos, levando a falhas de recepção de virtualmente todas as mensagens enviadas.

³ Representação visual do espectro de frequências de um sinal conforme ele varia com o tempo.

A sobreposição do sinal de um transmissor com outros sinais no meio de transmissão com frequências similares leva, após a realização de FFT para gerar o espectro Fourier, a um número maior do que o esperado (com base nos protocolos definidos) de barras de frequência com valor de amplitude significativo, inviabilizando a determinação das frequências corretas a serem decodificadas pelo receptor para obtenção de mensagem. A Figura 14 apresenta espectrogramas gerados a partir de uma instância do problema, com transmissões ocorrendo isoladamente e também de forma simultânea, situação em que pode-se observar claramente a sobreposição de frequências.

Figura 14 – Espectrogramas do sinal captado durante transmissão simultânea.



Fonte: Elaborada pelo autor.

5.5 Influência de Obstáculos Físicos

Para avaliar a influência de obstáculos físicos na transmissão, nós repetimos o experimento das seções 5.2 e 5.3 utilizando como obstrução física caixas de papelão posicionadas a um

metro de distância do dispositivo receptor, conforme retratado na Figura 15. No total, a obstrução era composta por quatro pilhas de caixas, cada uma com quatro caixas, em uma mesa diante do receptor. As caixas continham monitores LED de 23 polegadas em seu interior (modelo Dell P2317H) e dimensões de 59 cm x 37 cm x 17 cm.

Figura 15 – Organização de componentes para testes de obstrução física.

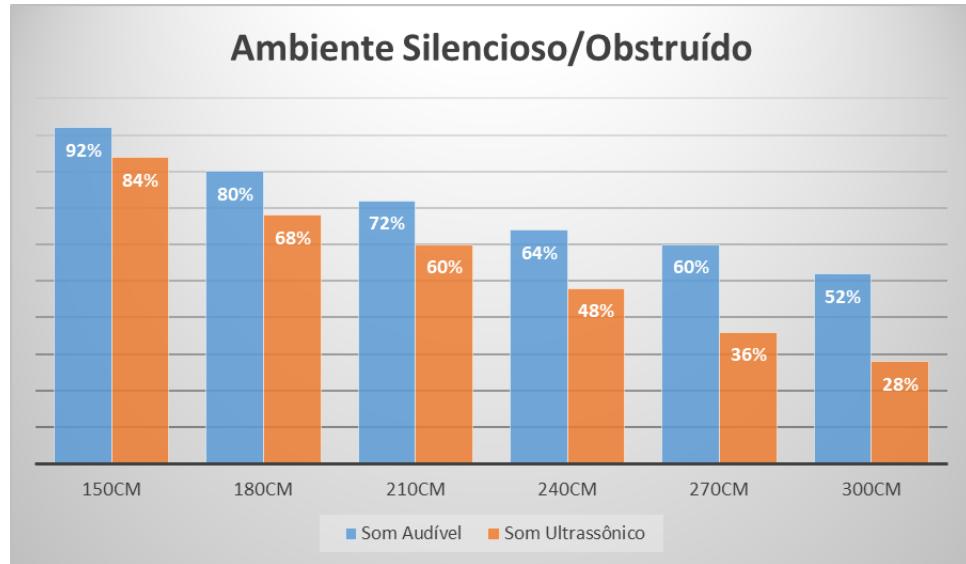


Fonte: Elaborada pelo autor.

Os gráficos da Figura 16 e da Figura 17 apresentam os resultados de percentual de sucesso de recepção de mensagens em relação à distância com presença de obstrução física nos cenário silencioso e ruidoso, respectivamente. A obstrução física introduzida entre transmissor e receptor no experimento teve efeito negativo considerável sobre a qualidade da comunicação nas distâncias testadas, com perdas de 20% ou mais das mensagens nas situações em que separamos os dispositivos por uma distância de pelo menos 180 cm. O protocolo ultrassônico foi especialmente afetado, com percentual de perdas significativamente maior em relação à sua contraparte audível em todas as distâncias testadas.

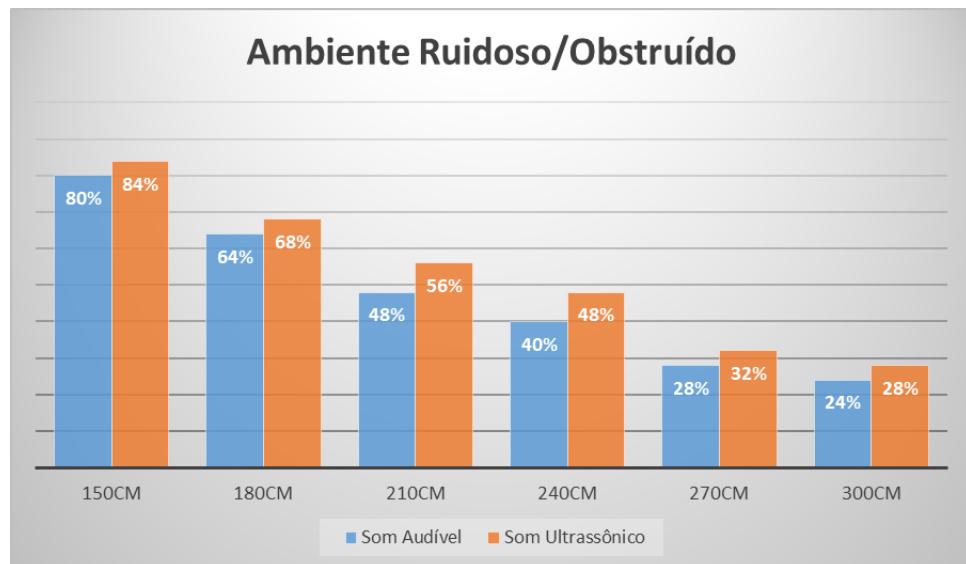
A atenuação de sinal causada pela fileira de pilhas de caixas foi suficiente para causar números elevados de perda de mensagens em todas as avaliações envolvendo distâncias superiores a 150 cm. O comprimento total da barreira física criada por todas as caixas (aproximadamente 240 cm), provavelmente contribuiu para a elevação da taxa de erros verificada em relação à distância, ao obstruir mesmo reflexões do sinal que poderiam ser captadas pelo microfone. Nós especulamos que o desempenho inferior produzido pelas transmissões ultrassônicas pode ser

Figura 16 – Taxa de sucesso na entrega de mensagens em relação à distância no cenário silencioso e com obstruções.



Fonte: Elaborada pelo autor.

Figura 17 – Taxa de sucesso na entrega de mensagens em relação à distância no cenário ruidoso e com obstruções.



Fonte: Elaborada pelo autor.

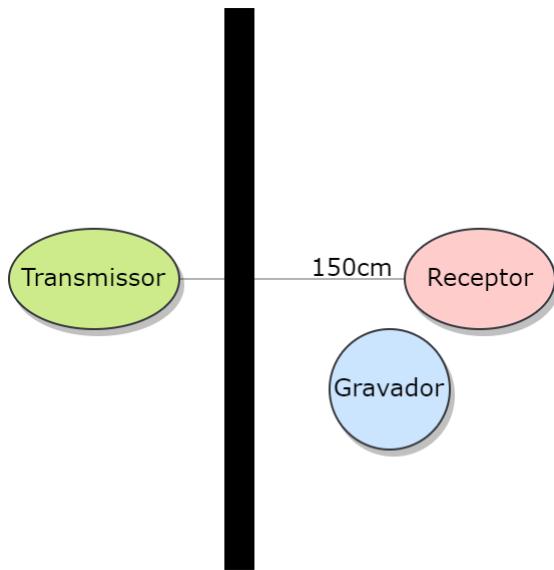
explicado pela influência da frequência de ondas sonoras em seu nível de atenuação por um material (WIKIBOOKS, 2020; PRITZ, 2004), com frequências mais baixas se propagando mais adiante por sofrerem menor absorção.

Em caráter complementar, realizamos testes para determinação do efeito de paredes como obstáculo físico na comunicação. O teste envolveu envio de mensagens entre um transmissor e um receptor em ambientes diferentes separados por uma parede de 15 centímetros de espessura.

Nós capturamos as transmissões com um smartphone (Asus ZenFone 5 T00J com Android

5.0) rodando seu aplicativo padrão de gravação de áudio e geramos espectrogramas das gravações com o editor de áudio Audacity. Devido à compressão com perda de dados nos arquivos gerados pelo aplicativo eliminarem áudio na faixa quase ultrassônica empregada pelos protocolos inaudíveis de ggwave, registramos apenas transmissões com protocolo audível para análise com espectrograma. A Figura 18 ilustra a disposição utilizada para os dispositivos no ambiente durante realização dos testes de comunicação através de parede.

Figura 18 – Representação da disposição de elementos utilizada no teste de comunicação simultânea.

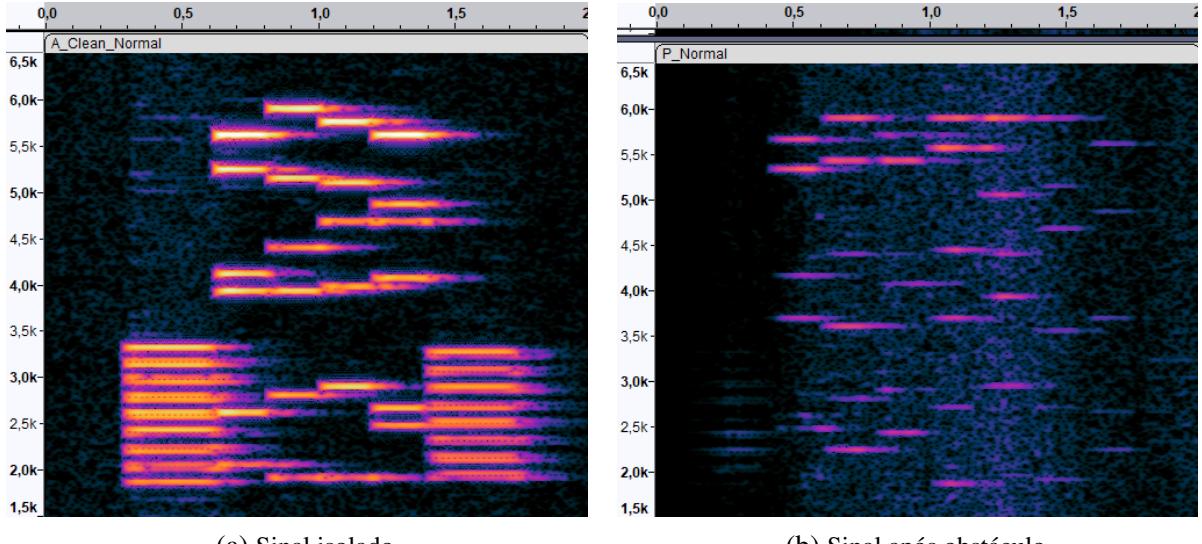


Fonte: Elaborada pelo autor.

Os testes realizados demonstraram que obstáculos de alta espessura e suficientemente densos como paredes podem ser capazes de bloquear quase completamente a propagação de ondas sonoras. Dentre as cinco instâncias do experimento realizadas, nenhuma delas apresentou recepção bem sucedida de mais do que duas das cinco mensagens transmitidas. As transmissões dos protocolos ultrassônicos, naturalmente mais susceptíveis a absorção pelo meio devido às altas frequências em que operam, foram completamente atenuadas pela parede, com nenhuma das tentativas de transmissão de mensagem (independentemente das configurações utilizadas) sendo bem sucedida.

A Figura 19 apresenta espectrogramas gerados a partir de uma instância da transmissão através de parede, com o sinal isolado e o mesmo sinal após passagem pelo obstáculo. Essa comparação evidencia a atenuação quase completa dos marcadores de inicio/fim da transmissão, necessários para detecção pelo receptor de que o áudio sendo captado pelo seu microfone contem uma mensagem a ser decodificada.

Figura 19 – Espectrogramas do sinal captado durante transmissão através de parede.



Fonte: Elaborada pelo autor.

5.6 Conclusão

Os experimentos que realizamos com ggwave demonstraram, em situações com ausência de obstáculos físicos ou ruído ambiente significativo, sua capacidade para transmissão de mensagens com taxa de sucesso total entre dispositivos separados por menos de três metros. Além dessa distância nós recomendamos, com base nos resultados, consideração adicional para assegurar que os equipamentos de áudio apresentam desempenho satisfatório em relação a faixa de frequências selecionada para comunicação.

Situações envolvendo ruído ambiente significativo testadas por nós resultaram em diminuição considerável do desempenho de ggwave, levando a perdas de mensagens do protocolo audível excedendo 10% nas distâncias avaliadas superiores a dois metros. Uso do protocolo ultrassônico se mostrou efetivo para contornar a interferência do ambiente ruidoso que criamos para o experimento, com resultados equivalentes aos medidos no cenário silencioso. Comparação dos espectros de frequência de gravações do cenário com fontes de ruído ligadas e desligadas evidenciou que a faixa de operação do protocolo ultrassônico permaneceu menos afetada (intensidade de frequências abaixo de -57 dB). Nós inferimos a partir desse resultado que, em situações de uso do ggwave que envolvem ruído ambiente com características similares ao produzido no ambiente de experimentação, emprego do protocolo ultrassônico pode solucionar problemas de interferência.

Um cenário alternativo de testes que realizamos para verificar o impacto de transmissões simultâneas resultou em falhas persistentes que inviabilizaram completamente a comunicação entre dispositivos. A geração de espectrogramas a partir das gravações dos testes que conduzimos

nos permitiu visualizar a sobreposição de frequências decorrente das transmissões simultâneas causadora das falhas no processo de decodificação de ggwave.

O cenário com obstruções físicas que criamos, envolvendo uma fileira de pilhas de caixas contendo monitores, se mostrou suficiente para redução drástica do desempenho de ggwave, com perdas de 20% ou mais de mensagens nas distâncias de pelo menos 180 cm avaliadas. O protocolo ultrassônico apresentou desempenho consideravelmente inferior ao protocolo audível nesse cenário quando ruído ambiente não estava também presente. Nós especulamos que tal diferença nos resultados pode ser atribuída à influência da frequência de ondas sonoras em seu nível de atenuação por um material (frequências mais baixas sofrerem menor absorção). Em situações com presença de obstáculos físicos de dimensões substanciais e similarmente densos aos empregados em nossos testes, nós recomendamos distâncias entre dispositivos não superiores a 150 cm para boa qualidade de comunicação.

A partir dos experimentos que conduzimos, nós inferimos que o ggwave demonstra potencial como protocolo de dados sobre som para emprego em cenários envolvendo comunicação de curto alcance entre dispositivos em ambientes fechados como escritórios ou laboratórios. Transmissões inaudíveis com ggwave se mostraram eficazes para comunicação em distâncias menores ou iguais a três metros na presença de ruído moderado, sendo esses resultados obtidos potencialmente representativos de situações reais que poderiam ser encontradas em tais ambientes. Os resultados também sugerem que obstáculos físicos densos e de grande extensão afetam consideravelmente as transmissões, de modo que atenção para manter o caminho entre dispositivos relativamente desobstruído pode ser necessária em cenários que façam uso do ggwave para atingir desempenho ideal.



PROTOCOLO DE TRANSMISSÃO POR PROXIMIDADE

6.1 Cenário

O estabelecimento de um protocolo de comunicação utilizando dados sobre som possibilita aplicações diversas. Rastreamento digital de contatos é um exemplo de cenário de uso.

Em linhas gerais, o rastreamento digital de contatos pode ser descrito como um caso particular do problema de detecção de proximidade (estimação da distância entre dois dispositivos), envolvendo seu emprego como solução de identificação e acompanhamento para quebra de cadeias de transmissão. Os dispositivos móveis usualmente geram identificadores que podem ser permanentes ou temporários, visando privacidade, e um servidor, que possui a lista com ambas as identidades de usuários reportando casos positivos, envia periodicamente dados anonimizados aos dispositivos para que estimem localmente o risco pessoal de exposição ([FLUERATORU et al., 2022](#)).

Dentre as estratégias empregadas nos protocolos das soluções disponíveis até o momento, BLE (*Bluetooth Low Energy*) tem sido o método dominante de rastreamento de proximidade ([O'NEILL; RYAN-MOSLEY; JOHNSON, 2020](#)). Apesar de suas vantagens em relação a métodos concorrentes como GPS, incluindo menor uso de bateria e estar menos sujeito a problemas de privacidade ([BAY et al., 2020](#)), existem preocupações em relação a eficácia de seu uso, especialmente em relação à potencial imprecisão envolvida no uso de sua força de sinal para detecção de situações de contato ([VAUGHN, 2020](#)).

Considerando as limitações encontradas nas soluções atualmente disponíveis, o uso de sinais ultrassônicos em rastreamento de contatos tem se mostrado promissor. As características físicas de ondas sonoras ultrassônicas, como o fato de que não se propagam através de paredes, permitem que soluções que as empregam reduzam ocorrência de falsos positivos ([LOH, 2020](#)).

Uma implementação básica de um sistema de rastreamento digital de contatos pode ser imaginada fazendo uso de pacotes de *advertising* para broadcasting de dados. Dispositivos móveis em posse de usuários poderiam alternar entre momentos em que realizam broadcasting de um valor identificador por meio de pacotes de *advertising* e momentos em que escaneiam o ambiente em busca de pacotes de *advertising* de dispositivos de usuários próximos. Um serviço de nuvem pode ser utilizado para armazenamento dos dados de identificação dos dispositivos, possibilitando acesso à base de dados de identificadores para determinar se um possível evento de contato ocorreu de acordo com os identificadores coletados por um dispositivo.

6.2 Transmissão por Proximidade no Bluetooth

Para oferecer um protocolo baseado em dados sobre som, utilizamos a inspiração do protocolo *Bluetooth Low Energy* ao projetar um protocolo de comunicação por proximidade. Esta seção apresenta a arquitetura do BLE e análise das diferenças entre as tecnologias Bluetooth e DoS.

6.2.1 *Bluetooth Low Energy*

O protocolo *Bluetooth Low Energy* foi introduzido com a versão 4.0 da especificação principal do Bluetooth ([GOMEZ; OLLER; PARADELLS, 2012](#)). Ele opera na banda de rádio de 2,4GHz entre 2400 MHz e 2483,5 MHz, dividindo-a em 40 canais com espaçamento de 2 MHz cada. Três desses canais são dedicados para disseminação de informações de dispositivos (*advertising*) e os 37 canais restantes são dedicados para propósito geral. Assim como o protocolo Bluetooth clássico, BLE emprega a técnica de *adaptive frequency hopping* para reduzir a chance de colisões durante transferências, através da seleção de um dos 37 canais para comunicação em certo intervalo de tempo, e utiliza o esquema de modulação FSK binário *Gaussian Frequency Shift Keying* (GFSK).

Na arquitetura Bluetooth, dispositivos podem assumir o papel de dispositivo central ou de dispositivo periférico. Desse modo, topologias estrela – em que um periférico realiza broadcasting de dados por meio de *advertising* – e ponto a ponto – em que há conexão de um dispositivo central com um ou mais periféricos – são possíveis.

6.2.2 *Protocolo de Advertisement*

O protocolo de *advertisement* é responsável pela publicação das informações de dispositivos disponíveis próximos e ativos, como identificação e tipo de dispositivo. Dispositivos periféricos podem transmitir cópias de pacotes *advertising* nos canais dedicados — um canal por vez em sequência aleatória — de tempos em tempos — momento conhecido como evento de *advertising* — utilizando atrasos aleatórios para evitar colisões persistentes com *advertisings* de outros dispositivos. Pacotes de *advertising* podem conter dados em um campo denominado

AdvData e o periférico pode potencialmente aceitar pacotes com requisição por dispositivos próximos para broadcast de dados adicionais complementares.

Para descobrir periféricos ao seu redor, dispositivos fazem uma varredura por pacotes *advertising* e, para requisitar uma conexão, respondem ao pacote de um deles no canal de *advertising*, levando-o a se tornar uma central (dispositivo mestre) caso a conexão seja aceita.

6.2.3 Protocolo de Estabelecimento de Conexão

Uma requisição de conexão enviada por um dispositivo para um periférico realizando *advertising* contém especificação de parâmetros, incluindo:

- **Endereço de acesso:** Valor alocado que identifica pacotes como pertencentes a uma certa conexão para permitir determinação por um receptor da relevância ou não de um pacote.
- **Intervalo de conexão:** Tempo entre cada evento de conexão, que é o momento em que central envia um pacote para iniciar trocas com periférico.
- **Latência do periférico:** Número de eventos de conexão consecutivos que periférico pode ignorar.
- **Timeout de supervisão:** Tempo máximo entre recebimento de dois pacotes para perda de ligação.
- **Mapa de canais:** Classificação de canais de propósito geral a se usar, que possibilita *frequency hopping*.

Com base nesses parâmetros, dispositivos periféricos sabem quando e em que canal eventos de conexão ocorrerão, possibilitando troca de dados com alternância entre transmissão e recepção certo número de vezes em cada evento de conexão após chegada do pacote inicial da central.

Pacotes de dados trocados entre dispositivos em uma conexão contêm campos de bit único chamados de SN e NESN que determinam como a troca de dados será controlada. A comunicação é iniciada com envio pela central (dispositivo mestre) de um pacote em que SN e NESN têm valor zero; pacote de resposta de um periférico têm NESN de valor um, o próximo pacote da central tem SN de valor um, o próximo pacote do periférico tem NESN de valor zero e assim sucessivamente. Um periférico sempre sabe o valor esperado do SN do próximo pacote a ser recebido e a central considera o recebimento de um pacote com NESN de valor igual ao valor do SN de seu próximo pacote como reconhecimento de uma transmissão correta. Um periférico assume que um pacote com SN de valor errado é uma retransmissão de um pacote anterior; a central reenvia um pacote com SN de mesmo valor se recebe um NESN de valor errado ou nenhuma resposta.

6.2.4 Análise Comparativa e de Viabilidade entre Bluetooth e Dados sobre Som

Considerando a viabilidade de implementação de versões análogas às características do *Bluetooth Low Energy* mencionadas nas seções anteriores utilizando dados sobre som, o principal desafio envolve a grande diferença de largura de banda disponível entre transmissões de rádio e a comunicação acústica.

BLE opera na banda de rádio de 2,4GHz entre 2400 MHz e 2483,5 MHz, dividindo-a em 40 canais com espaçamento de 2 MHz cada. Três desses canais são dedicados para *advertising* e os 37 canais restantes são dedicados para propósito geral, sendo utilizados com base no emprego de *frequency hopping* adaptativo para reduzir a chance de colisões durante transferências pelos dispositivos.

Comunicação via dados sobre som, por sua vez, precisa operar em uma faixa de frequências entre por volta de 200 Hz e 20000 Hz. Para comunicação ultrassônica, que é abordagem almejada para a vasta maioria das potenciais aplicações da tecnologia, a banda disponível é ainda mais restrita, limitando-se a aproximadamente 5 kHz com operação na faixa entre por volta de 15 kHz e 20 kHz. O growave faz uso de uma faixa de 4,5 kHz para transmissão de dados em seus protocolos (com variantes dual-tone de menor taxa de dados empregando uma faixa reduzida de 1,5 kHz disponíveis), possibilitando uma taxa de dados de até 16 Bps.

Dadas essas diferenças, a implementação de um modo de comunicação orientado a conexão correspondente ao modelo empregado pelo BLE via dados sobre som é inviabilizado. Uma conexão baseada na divisão do canal físico em unidades de tempo não sobrepostas (eventos de conexão) em que utiliza-se certa frequência de canal de dados e pacotes são trocados de foma alternada entre central e periférico em rápida sucessão não seria possível com comunicação acústica. A faixa de frequências disponível não é suficientemente ampla para ser dividida em múltiplos canais de comunicação e incapaz de suportar uma taxa de dados aceitável com tal esquema de controle de acesso ao meio.

Uma implementação análoga do protocolo de *advertisement* empregado pelo BLE utilizando dados sobre som é possível, mas uma abordagem alternativa seria necessária para conexões entre dispositivos se comunicando acusticamente.

6.3 Projeto do Protocolo

Esta seção descreve o projeto e implementação de um protocolo de transmissão por proximidade utilizando dados sobre som. Embora a sua implementação seja baseada em growave, ele poderia, em teoria, fazer uso de outro protocolo de camada física, como BLE. O protocolo implementado se situa na camada de enlace, sem entretanto fornecer serviços como entrega confiável, devido às limitações do growave.

6.3.1 Endereçamento

O protocolo identifica as estações por meio de *Universally Unique Identifiers* (UUIDs) da versão 4, na qual identificadores são gerados aleatoriamente. O UUID é um endereço de 128 bits que, para propósitos práticos, é único, sem necessidade de uma autoridade central de registros presente para garantir tal propriedade. Apesar da possibilidade de duplicação de um UUID existir, sua probabilidade é geralmente considerada negligenciável.

As mensagens trocadas apresentam um campo contendo o identificador de sua estação de origem, um campo contendo sua numeração e um campo contendo seu conteúdo. Mensagens de gwave têm um limite de 140 bytes de comprimento para o *payload* de cada transmissão, dos quais 36 bytes são consumidos pelo endereço de origem baseado em UUID, restando 104 bytes para os demais componentes das mensagens.

Para ser capaz de operar, uma rede de comunicação necessita de identificadores únicos associados às suas estações componentes. Dada a ausência de endereços MAC designados por seus fabricantes dentre os dispositivos de áudio envolvidos na comunicação via dados sobre som, nós utilizamos UUIDs em nosso protocolo para prover endereços físicos de rede para cada estação.

6.3.2 Estrutura de Mensagens

O protocolo serializa as mensagens a serem transmitidas pela camada física com Protocol Buffers ([GOOGLE, 2022](#)), um formato multiplataforma e de código aberto para serializar dados estruturados. Sua utilização envolve descrição das especificações de estruturas de dados desejadas em um arquivo de definição de mensagem que chamamos de Proto, a partir do qual o compilador disponível gera código para serializar e de-serializar na linguagem desejada passível de invocação por um remetente ou destinatário dessas estruturas.

A definição Proto criada para as mensagens contêm um campo de tipo *string* denominado *senderid* para inclusão do UUID da estação, um campo de tipo *int32* denominado *msgnum* para inclusão da numeração da mensagem e um campo de tipo *string* denominado *content* para inclusão do conteúdo da mensagem. O Código-fonte 3 apresenta o código do arquivo *.proto* de definição que produzimos para nossas mensagens.

Código-fonte 3 – Código contido em nosso arquivo *.proto* de definição

```
1: syntax = "proto2";
2:
3: package soundcomm;
4:
5: message Message {
6:     optional string senderid = 1;
```

```

7:     optional int32 msgnum = 2;
8:     optional string content = 3;
9: }
```

6.3.3 Processamento de Mensagens

O código que implementamos habilita a comunicação entre estações através da recepção e transmissão de mensagens. Ele atribui um identificador para cada estação, as quais alternam de tempos em tempos entre modo de recepção e modo de transmissão de mensagens. As mensagens recebidas levam a adição de seu identificador a uma lista atualizada regularmente das estações atualmente sob alcance. Um mecanismo de retransmissão de mensagens de estações alheias está presente para possibilitar a disseminação de mensagens além do alcance de transmissão de sua estação em um ambiente.

6.3.4 Implementação

A implementação foi feita em Python. O código de transmissão/recepção especifica uma classe com um método para transmissão e outro, para recepção de mensagens. Quando o código é inicializado em uma estação, os métodos são executados alternadamente. A alternância entre modo de transmissão e de recepção ocorre por meio da criação de duas threads, as quais se comunicam entre si com o uso de events do módulo `threading` de modo a executarem alternadamente os métodos citados.

Ao serem inicializadas, todas as estações entram primeiramente em modo de recepção. As estações permanecem nesse modo por um intervalo aleatório entre 15 e 30 segundos e então alternam para o modo de transmissão. Os textos produzidos como conteúdo para cada mensagem transmitida pelas estações no experimento realizado consiste em uma string de 18 bytes contendo um número real com ponto flutuante gerado aleatoriamente com o método `random()` do módulo `random`. Após o envio de uma mensagem, a estação retorna para o modo de recepção, no qual permanece durante um novo intervalo aleatório entre 15 e 30 segundos antes de retornar ao modo de transmissão.

Mensagens recebidas durante a execução do laço responsável pelo modo de recepção são adicionadas a uma lista acessível também pelo modo de transmissão. Quando a thread para transmissão é reativada, essa lista é verificada e, caso exista alguma entrada presente, sua última mensagem é removida e retransmitida pela estação em substituição a uma mensagem própria. Com isso, a mensagem em questão pode potencialmente se propagar para estações além do alcance de transmissão de sua estação de origem.

O Apêndice A contém o código do protocolo de transmissão.

6.4 Avaliação

Esta seção apresenta um experimento para avaliação da possibilidade de uso do nosso protocolo de dados sobre som para transmissão de mensagens por proximidade entre múltiplos dispositivos. O experimento avaliou a possibilidade de comunicação além do alcance de transmissão convencional de cada estação por meio de retransmissões de mensagens alheias pelas estações.

Nós utilizamos três computadores como estações receptoras/transmissoras de mensagens, posicionados de modo que o alcance de transmissão de cada um deles não cobre as demais máquinas confiavelmente. Assim, a retransmissão de mensagens entre estações se faz necessária para possibilitar disseminação de uma mensagem entre todos os computadores.

6.4.1 Ambiente de Experimentação

O experimento foi baseado na versão 0.4.2 do gwave através de seus *bindings* Python, em computadores com Python 3.10.4 e a biblioteca PyAudio 0.2.12¹. Nós executamos um código nos três computadores para habilitar a recepção e transmissão/retransmissão de mensagens. Observamos se estações, posicionadas fora do alcance de transmissão de uma mensagem, receberam tal mensagem através de retransmissão por sua estação mais próxima.

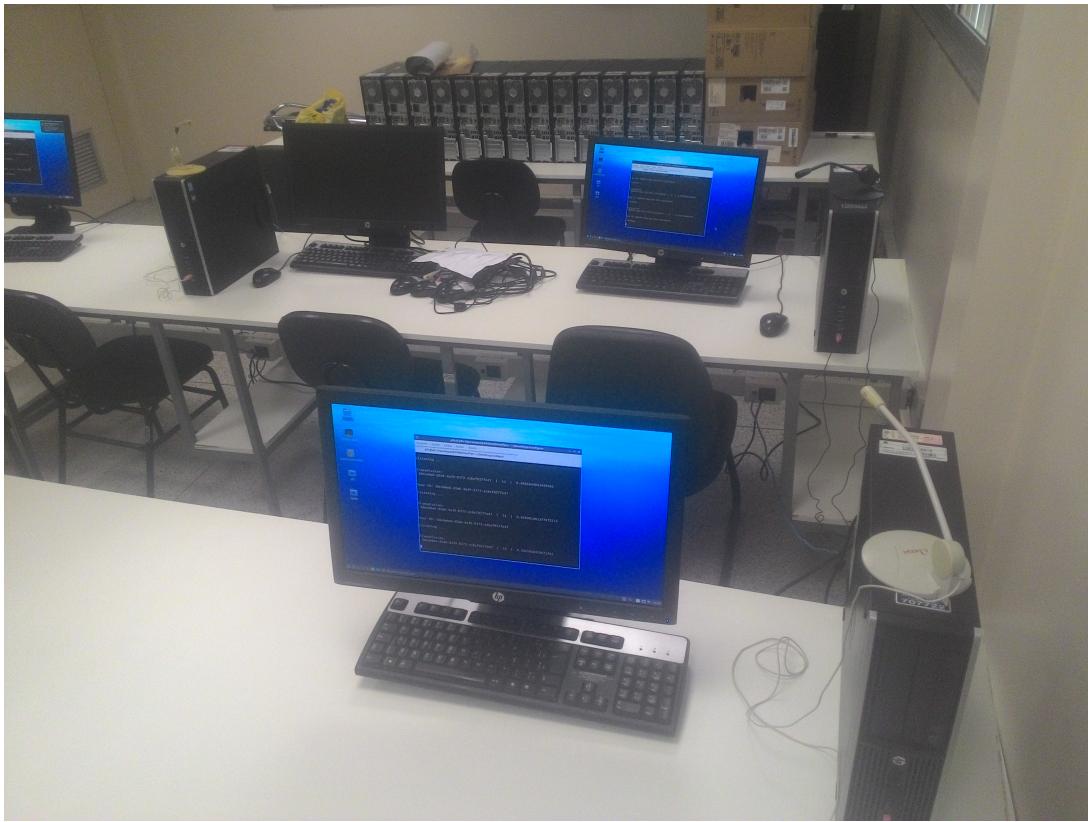
Devido à incapacidade de reprodução de frequências na faixa quase ultrassônica dos alto-falantes internos dos computadores utilizados, testamos apenas transmissões audíveis de gwave em seu protocolo padrão (`protocolId = 1`) no experimento. Uma faixa de frequências sonoras de 4,5 kHz é empregada para transmissões, tendo como frequência inicial 1875 Hz. A taxa de transferência de dados utilizada no protocolo é a configuração *Fast*, que corresponde a uma taxa de 16,76 B/s.

Nós conduzimos o experimento no Laboratório de Sistemas Digitais da Universidade Federal de Catalão (Sala 16 do Bloco J). Os dispositivos utilizados consistem em computadores HP Elite 8200 SFF conectados a microfones de mesa por cabos P2 estéreo de 3,5 mm, conforme a Figura 20. Suas especificações técnicas são listadas na Figura 9, sob "Computador receptor" e "Microfone". As transmissões ocorreram com os próprios alto-falantes internos das estações.

Para os testes, nós ajustamos a sensibilidade dos microfones de mesa de acordo com os níveis de som captados por eles no cenário silencioso do ambiente de experimentação, de modo a mantê-los todos abaixo de aproximadamente -30 dB, confortavelmente distantes do nível máximo que eles são capazes de reproduzir. Com esse objetivo, aplicamos uma atenuação de -18,06 dB aos microfones, o que correspondeu à configuração de 50% do volume da entrada de áudio dos computadores conectados. Configuramos o volume de 100% para os alto-falantes internos dos computadores.

¹ <<https://pypi.org/project/PyAudio/0.2.12/>>

Figura 20 – Organização de estações para experimento de disseminação de mensagens.



Fonte: Elaborada pelo autor.

6.4.2 Interferências

Em preparação para o experimento, visando determinar a viabilidade de disseminação de mensagens entre estações em um ambiente por meio de retransmissão, buscamos eliminar quaisquer fontes de interferência que poderiam afetar negativamente a realização do experimento. Nós posicionamos as estações de modo a assegurar que as duas estações mais distantes não conseguem transmitir diretamente mensagens uma para a outra. Além disso, nós removemos possíveis barreiras físicas entre elas e não introduzimos nenhuma fonte de ruído ambiente.

Ao conduzir o teste, a ausência de um mecanismo específico para mitigação de transmissões simultâneas no código que produzimos para as estações se mostrou problemática. Transmissões concorrentes de mensagens entre as estações ocorreram com frequência, interferência que levou a falhas de decodificação de grande parcela das mensagens e dificultou visualização da propagação de dados pelas estações do ambiente via retransmissões como almejado.

6.4.3 Discussão

Durante o experimento, ao definir individualmente a execução do modo de recepção ou de transmissão em cada estação, foi possível observar o recebimento das mensagens de uma estação, retransmitidas por sua estação vizinha, pela estação posicionada além de seu alcance

de transmissão. Entretanto, quando inicializamos todas as estações sem controle individual dos momentos em que cada uma se encontra em modo de transmissão ou recepção, transmissões simultâneas entre estações se tornou constante, levando a erros de decodificação persistentes e impedindo observação da disseminação de mensagens entre estações distantes desejada.

O método rudimentar de intervalos aleatórios para definição dos momentos de recepção e de transmissão de mensagem pelas estações se mostrou inadequado, resultando em erros de decodificação frequentes causados pela interferência gerada por transmissões simultâneas. O experimento que realizamos demonstrou a necessidade de um método mais sofisticado para escolha do momento em que estações transmitem mensagens, que busque detectar quando outras transmissões estão ocorrendo para reduzir substancialmente a ocorrência de colisões.

A designação de faixa de frequências específicas para cada estação que não se sobreponem em espaço de frequência potencialmente possibilitaria recepção de dados simultânea. Caso transmissões inaudíveis sejam desejadas, para acomodação de múltiplas estações dentre as frequências quase ultrassônicas reproduzíveis em hardware de áudio convencional, seria necessário empregar faixas de frequências reduzidas que diminuem consideravelmente a já limitada taxa de dados oferecida. Entretanto, tal abordagem não pode ser escalada para um maior número de estações. Uma solução escalável envolveria modificação mais extensa e fundamental do ggwave.

A adição de tratamento para duplicações de mensagens também é necessária. O código produzido não tem provisões para diferenciar potenciais mensagens duplicadas recebidas das estações no ambiente de comunicação. Cada nova mensagem precisa ser comparada a mensagens prévias recebidas pela estação e descartada caso seus atributos (identificador de origem, número e conteúdo) correspondam a uma delas.

O experimento que realizamos reforçou a viabilidade de disseminação de mensagens em um ambiente e da implementação de um protocolo de *advertisement* semelhante ao empregado pelo BLE com dados sobre som, considerando o broadcasting de dados bem sucedido observado entre as estações. Contudo, a faixa de frequências disponível para comunicação acústica, sendo vastamente inferior em relação a banda de rádio de 2,4 GHz, representa um obstáculo na criação de múltiplos canais de comunicação para mitigar problemas associados a transmissões simultâneas e para implementação de um modo de comunicação orientado a conexão como oferecidos pelo BLE.

6.5 Conclusão

Este capítulo explorou a implementação de um protocolo para comunicação por dados sobre som. Ele se situa na camada de enlace e é construído sobre o protocolo DoS de camada física ggwave.

Nós analisamos o funcionamento do *Bluetooth Low Energy* como fonte de inspiração para

produção de nosso protocolo DoS para comunicação entre estações. Conduzimos um experimento empregando tal protocolo para verificar a viabilidade de disseminação de mensagens em um ambiente via retransmissões e identificar potenciais limitações de nosso código.

O código que produzimos possibilitou a recepção de uma mensagem pelas estações fora do alcance de transmissão de sua estação de origem e reforçou a viabilidade da implementação de um protocolo de *advertisement* semelhante ao empregado pelo BLE com dados sobre som. Entretanto, nosso código não possui um método efetivo para reduzir a ocorrência de transmissões simultâneas entre as estações e a faixa de frequências disponível para comunicação acústica representa um obstáculo na criação de múltiplos canais de comunicação para mitigar tais problemas, bem como para implementação de um modo de comunicação orientado a conexão como oferecidos pelo BLE.

A partir dos resultados do experimento, nós concluímos que inclusão de um mecanismo sofisticado para redução de colisões em nosso protocolo que seja compatível com às limitações de banda de DoS, bem como adição de tratamento para duplicações de mensagens, são essenciais para suas aplicações futuras almejadas.



CONCLUSÃO

Este trabalho explorou a implementação de um protocolo para comunicação por dados sobre som. Nosso protocolo opera na camada de enlace e foi construído sobre o protocolo DoS de camada física gwave.

Nós conduzimos experimentos para avaliação do desempenho do protocolo de dados sobre som gwave. Os experimentos avaliaram o efeito de distância, ruído ambiente e obstáculos físicos na taxa de sucesso da transmissão de mensagens. Nos testes realizados, contabilizamos o número de mensagens recebidas corretamente por um dispositivo receptor em diferentes distâncias em relação ao receptor e de acordo com a presença ou ausência de obstrução física entre eles em uma sala fechada com ruído ambiente ou silenciosa.

Nós analisamos o funcionamento do *Bluetooth Low Energy* como fonte de inspiração para produção de nosso protocolo DoS para comunicação entre estações. Conduzimos um experimento empregando tal protocolo para verificar a viabilidade de disseminação de mensagens em um ambiente via retransmissões e identificar potenciais limitações de nosso código. O experimento avaliou a disseminação de mensagens entre estações, considerando que as duas estações mais distantes não conseguem transmitir diretamente uma para a outra, a partir de retransmissões de mensagens alheias pelas estações para possibilitar sua propagação pelo ambiente.

7.1 Contribuições

Os experimentos que realizamos com gwave demonstraram, em situações com ausência de obstáculos físicos ou ruído ambiente significativo, sua capacidade para transmissão de mensagens com taxa de sucesso total em distâncias menores que três metros. Além dessa distância nós recomendamos, com base nos resultados que obtivemos, consideração adicional para assegurar que os equipamentos de áudio empregados apresentam desempenho satisfatório em relação a faixa de frequências selecionada para comunicação.

Situações envolvendo ruído ambiente significativo resultaram em diminuição considerável de desempenho, levando a perdas de mensagens do protocolo audível excedendo 10% em distâncias superiores a dois metros. Uso do protocolo ultrassônico se mostrou efetivo para contornar a interferência do ambiente ruidoso que criamos, com resultados equivalentes aos medidos no cenário silencioso. Nós inferimos a partir desse resultado que, em situações de uso que envolvem ruído ambiente de característica similar, emprego do protocolo ultrassônico de ggwave pode solucionar problemas de interferência.

O cenário com obstruções físicas que criamos, envolvendo uma fileira de pilhas de caixas contendo monitores, se mostrou suficiente para redução drástica de desempenho, com perdas de 20% ou mais de mensagens em distâncias de pelo menos 180 cm. O protocolo ultrassônico apresentou desempenho consideravelmente inferior ao protocolo audível quando ruído ambiente não estava também presente. Nós especulamos que tal diferença pode ser atribuída à influência da frequência de ondas sonoras em seu nível de atenuação por um material. Em situações com presença de obstáculos físicos densos e de dimensões substanciais, nós recomendamos distâncias entre dispositivos de até 150 cm para boa qualidade de comunicação.

A partir dos experimentos que conduzimos, nós inferimos que ggwave demonstra potencial como protocolo de DoS para emprego em cenários envolvendo comunicação de curto alcance em ambientes fechados como escritórios ou laboratórios. Transmissões inaudíveis se mostraram eficazes para comunicação em distâncias menores ou iguais a três metros na presença de ruído moderado, sendo esses resultados obtidos potencialmente representativos de situações reais que poderiam ser encontradas em tais ambientes. Os resultados também sugerem que obstáculos físicos densos e de grande extensão entre os dispositivos comunicantes afetam consideravelmente as transmissões, de modo que atenção para manter o caminho entre dispositivos relativamente desobstruído pode ser necessária para atingir desempenho ideal.

O código que produzimos para teste de disseminação de mensagens em um ambiente com nosso protocolo possibilitou a recepção de uma mensagem pelas estações fora do alcance de transmissão de sua estação de origem e reforçou a viabilidade da implementação de um protocolo de *advertisement* semelhante ao empregado pelo BLE com dados sobre som. Entretanto, nosso código não possui um método efetivo para reduzir a ocorrência de transmissões simultâneas entre as estações e a faixa de frequências disponível para comunicação acústica representa um obstáculo na criação de múltiplos canais de comunicação para mitigar tais problemas, bem como para implementação de um modo de comunicação orientado a conexão como oferecidos pelo BLE.

7.2 Trabalhos Futuros

Testes mais extensos para avaliação do desempenho de ggwave em situações diversas são necessários para maior compreensão de suas capacidades e limitações. Experimentos de

comunicação conduzidos em ambientes abertos, com presença de ruído ambiente de nível e caráter variáveis, com tipos diferentes de obstruções físicas entre dispositivos e de determinação da distância máxima para transmissões efetivas estão entre as lacunas a serem preenchidas para maior compreensão do desempenho que pode ser esperado do protocolo em cenários reais de uso.

A partir dos resultados do experimento que conduzimos com nosso protocolo DoS, nós concluímos que inclusão de um mecanismo sofisticado para redução de colisões em nosso protocolo que seja compatível com às limitações de banda características de dados sobre som, bem como adição de tratamento para duplicações de mensagens, são essenciais para suas aplicações futuras almejadas.

A implementação de uma aplicação de demonstração (protótipo) é outro passo futuro necessário para validação do nosso protocolo. Um sistema básico de rastreamento digital de contatos empregando DoS pode ser imaginado, com pacotes de *advertising* para broadcasting de dados entre dispositivos móveis (os quais alternam entre momentos de broadcasting de um valor identificador e momentos em que escaneiam o ambiente por pacotes de outros dispositivos) e um serviço de nuvem armazenando os dados de identificação acessado pelos dispositivos para determinar ocorrência de eventos de contato de acordo com os identificadores que coletaram.

REFERÊNCIAS

- ASHIHARA, K. Hearing thresholds for pure tones above 16khz. **The Journal of the Acoustical Society of America**, AIP Publishing, v. 122, n. 3, p. EL52–EL57, 2007. Citado na página 11.
- BAY, J.; KEK, J.; TAN, A.; HAU, C. S.; YONGQUAN, L.; TAN, J.; QUY, T. A. Bluetrace: A privacy-preserving protocol for community-driven contact tracing across borders. **Government Technology Agency-Singapore, Tech. Rep**, v. 18, 2020. Citado na página 45.
- BUBLEY, D. Data over sound technology: device-to-device communications & pairing without wireless radio networks. **Disruptive Analysis Ltd**, p. 29, 2017. Citado na página 11.
- CADETE, J. Técnicas de conversão analógico-digital (a/d) / digital - analógico (d/a). **Redes de Comunicação 10º Informática**, 2010. Disponível em: <<https://rcjoaocadete.blogspot.com/2010/10/licao-n17-e-18-tecnicas-de-conversao.html>>. Acesso em: 8 mar. 2023. Citado na página 22.
- FLUERATORU, L.; SHUBINA, V.; NICULESCU, D.; LOHAN, E. S. On the high fluctuations of received signal strength measurements with ble signals for contact tracing and proximity detection. **IEEE Sensors Journal**, v. 22, n. 6, p. 5086–5100, 2022. Citado na página 45.
- GERASIMOV, V.; BENDER, W. Things that talk: Using sound for device-to-device and device-to-human communication. **IBM Systems Journal**, v. 39, n. 3.4, p. 530–546, 2000. Citado na página 15.
- GERGANOV, G. gwave. **gwave Github Project**, 2022. Disponível em: <<https://github.com/ggerganov/gwave>>. Acesso em: 17 mar. 2022. Citado nas páginas 20, 21 e 27.
- GOMEZ, C.; OLLER, J.; PARAELLS, J. Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. **Sensors**, v. 12, n. 9, p. 11734–11753, 2012. ISSN 1424-8220. Disponível em: <<https://www.mdpi.com/1424-8220/12/9/11734>>. Citado na página 46.
- GOOGLE. Protocol buffers. **Google Developers Site**, 2022. Disponível em: <<https://developers.google.com/protocol-buffers>>. Acesso em: 17 mar. 2022. Citado na página 49.
- JOSEPH-RAZ, L. Why it's so challenging to develop data over sound technology? **SONARAX Data Over Sound Blog**, 2020. Disponível em: <<https://www.sonarax.com/post/why-its-so-challenging-to-develop-data-over-sound-technology-why-the-other-protocols-failed>>. Citado na página 17.
- LIU, J.; CHEN, C.; MA, Y.; XU, Y. Energy analysis of device discovery for bluetooth low energy. In: IEEE. **2013 IEEE 78th Vehicular Technology Conference (VTC Fall)**. [S.I.], 2013. p. 1–5. Citado na página 11.
- LOH, P. Accuracy of bluetooth-ultrasound contact tracing: experimental results from novid ios version 2.1 using five-year-old phones. **NOVID**, 2020. Disponível em: <<https://www.novid.org/downloads/20200626-accuracy.pdf>>. Citado nas páginas 17 e 45.

LUBINETS, M. Reed solomon bch encoder and decoder. 2022. Disponível em: <<https://github.com/mersinvald/Reed-Solomon>>. Acesso em: 1 ago. 2022. Citado na página 29.

MARNEWICK, K.; NESFIELD, J.; MEHRABI, A.; JONES, D. **Why data-over-sound is an integral part of any IoT engineer's toolbox: Chirp + Arm = frictionless low power connectivity.** [S.l.], 2019. Disponível em: <<https://armkeil.blob.core.windows.net/developer/Files/pdf/white-paper/chirp-arm-data-over-sound.pdf>>. Citado nas páginas 11 e 14.

OOURA, T. General purpose fft (fast fourier/cosine/sine transform) package. 2006. Disponível em: <<https://www.kurims.kyoto-u.ac.jp/~ooura/fft.html>>. Acesso em: 1 ago. 2022. Citado na página 27.

O'NEILL, P.; RYAN-MOSLEY, T.; JOHNSON, B. A flood of coronavirus apps are tracking us. **MIT Technology Review**, 2020. Disponível em: <<https://www.technologyreview.com/2020/05/07/1000961/launching-mit-tr-covid-tracing-tracker/>>. Citado na página 45.

PERIYASAM, M.; DHANASEKARAN, R. Electromagnetic interference on critical medical equipments by rf devices. In: **2013 International Conference on Communication and Signal Processing**. [S.l.: s.n.], 2013. p. 78–82. Citado na página 15.

PRITZ, T. Frequency power law of material damping. **Applied Acoustics**, v. 65, n. 11, p. 1027–1036, 2004. ISSN 0003-682X. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S0003682X04000830>>. Citado na página 41.

QUIET. Quiet modem project. 2022. Disponível em: <<https://github.com/quiet/quiet>>. Acesso em: 16 mar. 2022. Citado na página 19.

QUIET-JS. Quiet-js: javascript binding for libquiet. 2022. Disponível em: <<https://github.com/quiet/quiet-js>>. Acesso em: 16 mar. 2022. Citado na página 19.

REED, I. S.; SOLOMON, G. Polynomial codes over certain finite fields. **Journal of the Society for Industrial and Applied Mathematics**, v. 8, n. 2, p. 300–304, 1960. Disponível em: <<https://doi.org/10.1137/0108018>>. Citado na página 29.

REN, Y.; WEN, P.; LIU, H.; ZHENG, Z.; CHEN, Y.; HUANG, P.; LI, H. Proximity-echo: Secure two factor authentication using active sound sensing. In: **IEEE. IEEE INFOCOM 2021-IEEE Conference on Computer Communications**. [S.l.], 2021. p. 1–10. Citado na página 16.

ROSEN, S.; HOWELL, P. **Signals and systems for speech and hearing**. [S.l.]: Brill, 2011. v. 29. 163 p. Citado na página 13.

SATOH, H.; SUZUKI, M.; TAHIRO, Y.; MORIKAWA, H. Ambient sound-based proximity detection with smartphones. In: **Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems**. [S.l.: s.n.], 2013. p. 1–2. Citado na página 16.

THIEL, B.; KLOCH, K.; LUKOWICZ, P. Sound-based proximity detection with mobile phones. In: **Proceedings of the Third International Workshop on Sensing Applications on Mobile Phones**. [S.l.: s.n.], 2012. p. 1–4. Citado na página 16.

VAUGHN, A. Bluetooth may not work well enough to trace coronavirus contacts. **New Scientist**, 2020. Disponível em: <<https://www.newscientist.com/article/2243137-bluetooth-may-not-work-well-enough-to-trace-coronavirus-contacts/>>. Citado na página 45.

WECKBACH, V. Astromech. **AstroMech Github Project**, 2022. Disponível em: <<https://github.com/weckbach/AstroMech>>. Acesso em: 17 mar. 2022. Citado na página 19.

WIKIBOOKS. **Engineering Acoustics/Outdoor Sound Propagation — Wikibooks, The Free Textbook Project**. 2020. [Online; accessed 14-July-2023]. Disponível em: <https://en.wikibooks.org/w/index.php?title=Engineering_Acoustics/Outdoor_Sound_Propagation&oldid=3781234>. Citado na página 41.

ZEPPELZAUER, M.; RINGOT, A. **SoniTalk: An Open Protocol for Data-Over-Sound Communication**. [S.l.], 2019. Disponível em: <<https://www.ietf.org/archive/id/draft-zeppelzauer-data-over-sound-00.txt>>. Acesso em: 16 de março de 2022. Citado na página 20.

—

CÓDIGO DE TRANSMISSÃO E RECEPÇÃO DE MENSAGENS

Código-fonte 4 – Código para transmissão/recepção de mensagens

```
1: import ggwave
2: import pyaudio
3: import messages_pb2
4: import random
5: import time
6: import uuid
7: import threading
8:
9: id = uuid.uuid4()
10:
11: p2 = pyaudio.PyAudio()
12:
13: stream2 = p2.open(format=pyaudio.paFloat32, channels=1, rate
   =48000, input=True, frames_per_buffer=1024)
14:
15: rtxlist = []
16:
17: class rtx:
18:
19:     def tx(self):
20:         while True:
21:             event_object.wait()
22:             txmessagelist = []
23:             z = 0
```

```
24:     while True:
25:         while rtxlist:
26:             p1 = pyaudio.PyAudio()
27:             waveform = ggwave.encode(rtxlist[-1].
28:             SerializeToString().decode(), protocolId = 1, volume = 25)
29:             print ('\nRetransmission:\n',uuid.UUID(rtxlist[-1].
30:             senderid)," | ",str(rtxlist[-1].msgnum)," | ",rtxlist[-1].
31:             content,' \n')
32:             stream1 = p1.open(format=pyaudio.paFloat32, channels
33: =1, rate=48000, output=True, frames_per_buffer=4096)
34:             stream1.write(waveform, len(waveform)//4)
35:             time.sleep(0.5)
36:             rtxlist.pop()
37:             p1.terminate()
38:             event_object.set()
39:             event_object.clear()
40:             event_object.wait()
41:             p1 = pyaudio.PyAudio()
42:             txmessage = messages_pb2.Message()
43:             txmessage.senderid = str(id)
44:             z = len(txmessagelist)
45:             txmessage.msgnum = z
46:             txmessage.content = str(random.random())
47:             txmessagelist.append(txmessage)
48:             waveform = ggwave.encode(txmessagelist[z].
49:             SerializeToString().decode(), protocolId = 1, volume = 25)
50:             print ('\nTransmission:\n',uuid.UUID(txmessagelist[z].
51:             senderid)," | ",str(txmessagelist[z].msgnum)," | ",
52:             txmessagelist[z].content,' \n')
53:             stream1 = p1.open(format=pyaudio.paFloat32, channels=1,
54:             rate=48000, output=True, frames_per_buffer=4096)
55:             stream1.write(waveform, len(waveform)//4)
56:             time.sleep(0.5)
57:             p1.terminate()
58:             event_object.set()
59:             event_object.clear()
60:             event_object.wait()

54:     def rx(self):
55:         while True:
56:             t = time.time() + random.randint(15, 30)
57:             p2 = pyaudio.PyAudio()
```

```
58:
59:     stream2 = p2.open(format=pyaudio.paFloat32, channels=1,
60:                         rate=48000, input=True, frames_per_buffer=1024)
61:     print ('\nYour ID:', id, '\n\nListening ... \n')
62:     instance = ggwave.init()
63:
64:     try:
65:         rxmessagelist = []
66:         rxidlist = {}
67:         w = 0
68:         while True:
69:             for x in rxidlist:
70:                 if float(rxidlist[x]) <= time.time():
71:                     print ('\nTransmitter list updated:', uuid.UUID(x),
72:                           'device ID removed\n')
73:                     rxidlist.pop(x)
74:                     break
75:             data = stream2.read(1024, exception_on_overflow=False
76: )
77:             res = ggwave.decode(instance, data)
78:             if (not res is None):
79:                 try:
80:                     try:
81:                         rxmessage = messages_pb2.Message()
82:                         rxmessage.ParseFromString(res)
83:                         w = len(rxmessagelist)
84:                         rxmessagelist.append(rxmessage)
85:                         print ('\nSender ID:', uuid.UUID(rxmessagelist[w
86: ].senderid), '\nMessage number:', r xmessagelist[w].msgnum, '\
87: nReceived text:', r xmessagelist[w].content, '\n')
88:                         if r xmessagelist[w].senderid not in rxidlist:
89:                             rxidlist.update({r xmessagelist[w].senderid:
90: str(time.time() + 60)})
91:                             print ('\nTransmitter list updated:', uuid.UUID
92: (r xmessagelist[w].senderid), 'device ID added\n')
93:                             rtxlist.append(r xmessagelist[w])
94:                         else:
95:                             rxidlist.update({r xmessagelist[w].senderid:
96: str(time.time() + 60)})
97:                             rtxlist.append(r xmessagelist[w])
98:                         except:
```

```
92:             print ('\nReceived text:', res.decode("utf-8"), '\n')
93:         except:
94:             pass
95:         if time.time() >= t:
96:             stream2.stop_stream()
97:             stream2.close()
98:             event_object.set()
99:             event_object.clear()
100:            event_object.wait()
101:            t = time.time() + random.randint(15, 30)
102:            stream2 = p2.open(format=pyaudio.paFloat32,
103:                               channels=1, rate=48000, input=True, frames_per_buffer=1024)
104:            print ('\nYour ID:', id, '\n\nListening ... \n')
105:        except KeyboardInterrupt:
106:            pass
107:        ggwave.free(instance)
108:
109:        stream2.stop_stream()
110:        stream2.close()
111:
112:        p2.terminate()
113:
114: class_obj = rxtx()
115:
116: if __name__ == '__main__':
117:     event_object = threading.Event()
118:
119: T1 = threading.Thread(target=class_obj.tx)
120: T2 = threading.Thread(target=class_obj.rx)
121:
122: T1.start()
123: T2.start()
```
