UNIVERSIDADE FEDERAL FLUMINENSE

DIOGO DE CASTRO PERDOMO

Uma Arquitetura Georreferenciada para a Seleção de Fluxos de Vídeo no Contexto de IoMT

NITERÓI

2019

UNIVERSIDADE FEDERAL FLUMINENSE

DIOGO DE CASTRO PERDOMO

Uma Arquitetura Georreferenciada para a Seleção de Fluxos de Vídeo no Contexto de IoMT

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: Sistemas de Computação

Orientador:
Prof. Dr. José Viterbo Filho

Co-orientadora:
Prof.ª Dra. Débora Christina Muchaluat Saade

NITERÓI

2019

Ficha catalográfica automática - SDC/BEE
Gerada com informações fornecidas pelo autor

P433a Perdomo, Diogo de Castro
Uma Arquitetura Georreferenciada para a Seleção de Fluxos de Vídeo no Contexto de IoMT / Diogo de Castro Perdomo ; José Viterbo Filho, orientador ; Débora Christina Muchaluat Saade, coorientadora. Niterói, 2019.
84 f. : il.

Dissertação (mestrado)-Universidade Federal Fluminense, Niterói, 2019.

DOI: http://dx.doi.org/10.22409/PGC.2019.m.04802600607

1. Serviço baseado em localização (LBS). 2. Internet das Coisas Multimídia (IoMT). 3. Câmeras IP. 4. Áreas de cobertura. 5. Produção intelectual. I. Filho, José Viterbo, orientador. II. Muchaluat Saade, Débora Christina, coorientadora. III. Universidade Federal Fluminense. Instituto de Computação. IV. Título.

CDD -

Bibliotecária responsável: Fabiana Menezes Santos da Silva - CRB7/5274

Diogo de Castro Perdomo

Uma Arquitetura Georreferenciada para a Seleção de Fluxos de Vídeo no Contexto de IoMT

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: Sistemas de Computação

Aprovada em Abril de 2019.

BANCA EXAMINADORA

Prof. José Viterbo Filho - Orientador, UFF

Profa. Débora Christina Muchaluat Saade - Coorientadora, UFF

Profa. Flavia Cristina Bernardini, UFF

Prof. Rodrigo Pereira dos Santos, UNIRIO

Niterói

2019

"A mente que se abre a uma nova idéia, jamais voltará ao seu tamanho original."

Albert Einstein

Agradecimentos

A presente dissertação de mestrado não poderia ser concluída sem o apoio de várias pessoas. Em primeiro lugar gostaria de agradecer aos meus pais Eloide Perdomo e Maria Teresa de Castro Perdomo por me darem apoio desde o início da graduação até o presente momento, incluindo os parentes próximos por me darem suporte.

Agradeço a ajuda e incentivos dos professores e em especial ao orientador José Viterbo Filho e a coorientadora Débora Christina Muchaluat Saade por estarem sempre dispostos a me corrigir e indicar os melhores caminhos. Também aos amigos que dispensaram tempo para ajudar nos testes em especial ao aluno Patrick Barreto dos Santos.

Por fim, apresento meus agradecimentos a CAPES pelo apoio financeiro, sem o qual não teria sido possível desenvolver esta pesquisa.

Muito obrigado a todos.

Resumo

Com o crescimento significativo de dispositivos multimídia com acesso à Internet e ubiquamente instalados, tornou-se possível implementar serviços para atender diferentes demandas da sociedade, como aquelas aplicadas à segurança, permitindo vislumbrar oportunidades para novas aplicações e serviços em direção a Internet das coisas multimídia (IoMT). Câmeras IP são exemplos comuns de dispositivos para esta finalidade que têm se tornado um item cada vez mais popular em centros urbanos. Também fazem parte do nosso cotidiano os serviços baseados em localização (LBS - *Location-Based Services*), auxiliando em diversas tarefas como traçar melhores caminhos para um destino, verificação de trânsito e mesmo pra aplicações de segurança como o monitoramento de automóveis, transporte de cargas ou de pessoas. No entanto, não é possível identificar visualmente algumas atividades como tentativa de roubo ou seqüestro como é feito por sistemas vigilância com câmeras. Sistemas de videovigilância com várias telas são geralmente usados por operadores os quais podem identificar situações, mas isso pode exigir muito esforço para identificar e rastrear alguma entidade por várias câmeras e suas respectivas telas. Foi identificado que a integração dessas duas informações (Posição Global e Vídeo) poderiam melhorar os sistemas de vigilância a partir de uma perspectiva de segurança, permitindo que a entidade monitorada possa ser acompanhada visualmente por câmeras ao longo do seu trajeto com base na sua localização por coordenadas geográficas tornando o monitortamento visual mais automatizado. Este trabalho propõe uma arquitetura georreferenciada para seleção de fluxos de vídeo para monitorar entidades (pessoas ou não) em um contexto de Internet de Coisas Multimídia (IoMT) através de um modelo representacional de áreas de cobertura de câmeras IP incluindo uma abordagem para o armazenamento e consulta de forma dinâmica para os metadados das câmeras, e um serviço de transmissão de vídeo para a Web. Um protótipo foi implementado para permitir a realização de uma prova de conceito da arquitetura proposta. Esse protótipo inclui um serviço de transmissão de vídeo de baixa latência para iniciar rapidamente uma transmissão de vídeo. Os testes foram realizados na Internet para verificar a latência do fornecimento de vídeo e o consumo de largura de banda. O cenário de testes compreendeu em uma câmera IP, um servidor de vídeo, um servidor de monitoramento, um servidor web e uma entidade enviando a sua localização. Os testes comprovaram a eficácia da proposta e sua baixa latência na transmissão de vídeo.

Palavras-chave: Serviço baseado em localização (LBS), Internet das Coisas Multimídia (IoMT), Câmeras IP, Áreas de cobertura, Segurança, Sistema de Posicionamento Global (GPS).

Abstract

With the significant growth of multimedia devices with Internet access and ubiquitously installed, it became possible to implement services to meet different demands of society, such as those applied to security, allowing to glimpse opportunities for new applications and services towards the Internet of multimedia things (IoMT). IP cameras are common examples of devices for this purpose that have become an increasingly popular item in urban centers. Location-Based Services (LBS) is also part of the daily routine, assisting in a variety of tasks such as tracing better paths to a destination, traffic verification, and even security applications such as car monitoring, transportation cargo or people. However, it is not possible to visually identify some activities such as attempted theft or kidnapping as is done by surveillance systems with cameras. Video surveillance systems with multiple screens are generally used by operators who can identify situations, but this may require a lot of effort to identify and track some entity through multiple cameras and their respective screens. It was identified that the integration of these two information (Global Position and Video) could improve surveillance systems from a security perspective, allowing the monitored entity to be monitored visually by cameras along its route based on its location by geographical coordinates, making visual monitoring more automated. This work proposes a georeferenced architecture for video stream selection to monitor entities (people or not) in a context of Internet Multimedia Things (IoMT) through a representational model of coverage areas of IP cameras including an approach to dynamic storage and query for camera metadata, and a web video streaming service . A prototype was implemented to allow a proof of concept of the proposed architecture. This prototype includes a low-latency video transmission service to quickly start a video transmission only when an entity is within a visual field of view. The tests were conducted on the Internet to check the latency of video delivery and bandwidth consumption. The test scenario comprised an IP camera, a video server, a monitoring server, a web server and an entity sending its location. The tests proved the effectiveness of the proposal and its low latency in video transmission.

Keywords: Location-based service (LBS), Internet of Multimedia Things (IoMT), IP cameras, Areas of coverage, Security, Global Positioning System (GPS), Web video transmission.

Lista de Figuras

1.1 Abordagem multi-metodológica para pesquisa de SI. Imagem baseada em [42] 5

2.1 Exemplo de modelos de representação de dados geoespaciais na forma Vetorial e Raster. 8

2.2 Representação de longitude e latitude. Fonte: desktop.arcgis.com 10

2.3 Exemplo com Leaflet 13

2.4 Exemplo com Mapbox 13

2.5 Exemplo com Openlayers 13

2.6 Exemplo de figuras geométricas no Google Maps. 14

2.7 Exemplo de fluxo de vídeo contínuo armazenado 16

2.8 Exemplo de voz e vídeo interativos 16

2.9 Exemplo de transmissão de fluxo contínuo ao vivo 16

2.10 Visão global do *HTTP Live Streaming* (HLS) 18

4.1 Arquitetura de monitoramento 31

4.2 Etapas dos módulos 32

4.3 Registro de metadados 32

4.4 Consulta de metadados 33

4.5 Consulta de metadados 34

4.6 Definição de área de cobertura 34

4.7 Determinação da área de cobertura através de um polígono. 35

4.8 Várias instâncias interligando logicamente o cliente à camera 36

4.9 Modelo onde é necessário apenas uma instância para n clientes. 37

5.1 Formato Binary JSON (BSON) 41

5.2 Exemplo de documentos GeoJSON para o tipo ponto e polígono 41

5.3 Representação de documento e interceptado por uma coordenada geográfica. 42

5.4 Servidor WebSocket recebendo mensagens e realizando pesquisas 43

5.5 Cabeçalho teórico Ogg 46

5.6 Cabeçalho de um fluxo real 47

5.7 Esquema atualizado para servidor de vídeo 47

5.8 Exemplo de edição. a) Ferramenta de desenho Drawing Control. b) Inserção de marcador e vértices do polígono. c) Marcador e polígono na camada de Dados 48

5.9 Novo controle para adicionar câmera e área de cobertura. 49

5.10 Procura por endereço no mapa 49

5.11 Desenho da área de cobertura 49

5.12 Formulário para preenchimento de metadados da câmera 50

5.13 Metadados registrados 50

5.14 a) Elemento do tipo Point representando a posição da câmera no mapa. (b) Polígono representando a área de cobertura onde cada vértice possui uma coordenada de latitude e longitude. c) Junção dos dois objetos. 51

6.1 Cenário de monitoramento 53

6.2 Envio de informações de monitoramento para a interface de usuário (cliente) 53

6.3 Metadados enviados ao cliente (Interface do usuário) 54

6.4 Funcionamento do servidor de vídeo desenvolvido 54

6.5 Retorno de elemento através de consulta com HTTP 55

6.6 Parâmetro Excellent (Teste de consumo de banda para transmissão de vídeo no formato MJPEG diretamente para a página HTML). 56

6.7 Parâmetro Good (Teste de consumo de banda para transmissão de vídeo no formato MJPEG diretamente para a página HTML). 57

6.8 Parâmetro Medium (Teste de consumo de banda para transmissão de vídeo no formato MJPEG diretamente para a página HTML). 57

6.9 Parâmetro Excellent (Teste de consumo de banda para a transcodificação de vídeo no formato MP4 para Theora). 58

6.10 Parâmetro Good (Teste de consumo de banda para a transcodificação de vídeo no formato MP4 para Theora). 58

6.11 Parâmetro Medium (Teste de consumo de banda para a transcodificação de vídeo no formato MP4 para Theora). 59

Lista de Tabelas

2.1 Exemplos de alguns softwares de GIS 9

2.2 Tipos de elementos do OSM 12

2.3 Tabela de ferramentas do Google Maps 13

2.4 Tabela de ferramentas do Google Maps 14

2.5 Comparativo de utilização de plataformas 15

3.1 Comparativos 28

5.1 Formato do documento 40

5.2 Exemplos de operadores de pesquisa 42

5.3 Exemplos de softwares para *streaming* de vídeo 45

5.4 Formatos suportados pela tag <video> 45

6.1 Configuração do Servidor Web e Servidor de Monitoramento (Mesma máquina) 59

6.2 Configuração do Servidor de Vídeo (Máquina Virtual) 59

6.3 Configuração da máquina Cliente 60

6.4 Configuração da Câmera IP 60

6.5 Média de bits por segundo para parâmetros de qualidade 60

6.6 Atrasos na recepção das imagens, n=10 (amostras) 61

Lista de Abreviaturas e Siglas

AAC	:	Advanced Audio Coding;
API	:	Application Programming Interface;
BSON	:	Binary JSON;
CDN	:	Content Delivery Network ou Content Distribution Network;
CFTV	:	Circuito Fechado de TV;
DD	:	Decimal degrees;
DMS	:	Degrees-minutes-seconds;
E911	:	Enhanced 911;
FLV	:	Enhanced 911;
FoV	:	Field of View;
GIS	:	Geographic Information Systems;
GPS	:	Global Positioning System;
HLS	:	HTTP Live Streaming;
HTML	:	HyperText Markup Language;
HTTP	:	Hypertext Transfer Protocol;
ICE	:	Interactive Connectivity Establishment;
IETF	:	Internet Engineering Task Force;
IoMT	:	Internet of Multimedia Things;
IoT	:	Internet of Things;
IP	:	Internet Protocol;
JSON	:	JavaScript Object Notation;
KML	:	Keyhole Markup Language;
LBS	:	Location-Based Services;
MJPG	:	Location-Based Services;
MPD	:	Media Presentation Description;
MPEG	:	Motion JPEG;
MPEG-DASH	:	MPEG-Dynamic Adaptive Streaming over HTTP;
NoSQL	:	MPEG-Dynamic Adaptive Streaming over HTTP;
NTP	:	Network Time Protocol;

Lista de Abreviaturas e Siglas

OdbL	:	Open Database License;
OSM	:	OpenStreetMap;
P2P	:	Peer-to-peer;
PoE	:	Power over Ethernet;
REST	:	Representational State Transfer;
RTCP	:	Real-Time Transport Control Protocol;
RTP	:	Real-Time Transport Protocol;
RTSP	:	Real Time Streaming Protocol;
SDK	:	Software Development Kit;
STUN	:	Session Traversal Utilities for NAT;
SVD	:	Sistema de Vídeo Desenvolvido;
TCP	:	Transmission Control Protocol;
TM2	:	2-degree Transverse Mercator;
TS	:	transport stream;
TURN	:	Traversal Using Relay NAT;
UDP	:	User Datagram Protocol;
UFF	:	Universidade Federal Fluminense;
URI	:	Uniform Resource Identifier;
URL	:	Uniform Resource Locator;
UTM	:	Universal Transverse Mercator;
VoIP	:	Voice over Internet Protocol;
W3C	:	World Wide Web Consortium;
Web	:	World Wide Web;
WGS84	:	World Geodetic System;

Sumário

1 Introdução **1**

1.1 Definição do problema 2

1.2 Objetivos 4

1.3 Metodologia 4

1.4 Organização do texto 6

2 Conceitos Básicos **7**

2.1 Geographic Information System (GIS) 7

2.2 Web Mapping 10

2.2.1 OpenStreetMap - OSM 10

2.2.2 Google Maps 13

2.2.3 Comparação entre as Plataformas 15

2.3 Vídeo Streaming na Internet 15

3 Trabalhos Relacionados **20**

3.1 Serviços baseados em localização (LBS) no contexto de monitoramento . . 20

3.2 Arquitetura IrisNet 21

3.3 Abordagem de Monitoramento Ubíquo Georreferenciado 22

3.4 Integração de Dispositivos Baseada em Serviços Web 23

3.5 Sistema de Vigilância Baseado em Contexto 24

3.6 IPTV em um contexto de Vigilância por Vídeo 24

3.7 Computação visual para o reconhecimento de placas 25

3.8 Centros de Comando e Controle 26

3.8.1 Discussão 28

4 Arquitetura Proposta 30

4.1 Visão Geral da Arquitetura 30

4.2 Servidor de Monitoramento 31

4.2.1 Módulo de Registro 32

4.2.2 Módulo de Consulta 33

4.2.3 Módulo de Monitoramento 33

4.3 Informações sobre a câmera e área de cobertura 34

4.4 Servidor de vídeo 36

4.5 Interface do usuário 38

5 Protótipo 39

5.1 Servidor de monitoramento 39

5.1.1 Módulo de registro 39

5.1.2 Módulo de consulta 41

5.1.3 Módulo de monitoramento 42

5.2 Servidor de vídeo 43

5.3 Interface do usuário 47

5.3.1 Desenho da área de cobertura 47

5.3.2 Cadastro da área de cobertura 51

6 Avaliação Experimental 52

6.1 Cenário 52

6.2 Análise dos Resultados 60

7 Conclusão 62

7.1 Trabalhos Futuros 63

Referências **65**

Capítulo 1

Introdução

Uma rede global de objetos exclusivamente endereçados com padrões de protocolos de comunicação é chamado pela semântica original [4] de Internet das Coisas ou IoT (*Internet of Things*), no qual os objetos interconectados aumentam a ubiquidade da Internet através de uma interação por sistemas embarcados que se comunicam entre si e com os seres humanos por uma rede altamente distribuída [52].

Sensores de vídeo como câmeras IP instalados ubiquamente podem se destacar no contexto de segurança no cenário de cidades inteligentes. O termo original de computação ubíqua foi cunhado por Mark Weiser em 1988 na Xerox PARC. Ele imaginou um futuro em que as tecnologias da computação se incorporassem aos artefatos do dia a dia, usadas para apoiar as atividades diárias ou igualmente aplicáveis ao nosso trabalho, ao gerenciamento de nossas casas e à diversão [29]. Segundo Weiser as tecnologias mais profundas e duradouras são aquelas que desaparecem. Elas dissipam-se nas coisas do dia a dia até tornarem-se indistiguíveis.

As informações do ambiente físico que são coletadas pelos dispositivos sensores de IoT (Internet of Things) tradicionais podem incluir temperatura, pressão, brilho, etc., e também o envio de dados de estado, como o nível da bateria. Para economizar recursos de memória e energia, estas aplicações exigem taxas de dados menores e usam ciclos (forma periódica) para envio dos dados.

Dispositivos com propriedades de multimídia podem ser incorporados à IoT para criar novas aplicações e serviços que inspirem uma visão da Internet de Coisas Multimídia (IoMT) [1]. Nesse caso, objetos multimídia inteligentes podem interagir uns com os outros e com os de outras categorias na Internet. Os dados multimídia da IoMT são volumosos por natureza e, em comparação com os dispositivos sensores tradicionais a comunicação

exige mais recursos de processamento e memória. Por exemplo, dados visuais de uma câmera de vigilância podem inquestionavelmente produzir ordens de grandeza de dados mais rapidamente do que dados produzidos por simples sensores não-multimídia.

As câmeras podem monitorar o fluxo do tráfego de automóveis, observar a vida selvagem, as condições climáticas, detectar intrusos, etc. Milhões de câmeras estão conectadas à Internet para uma variedade de propósitos [28]. O tráfego global de vídeo na Internet será de até 82% em 2022 e dentre este percentual o tráfego relacionado à videovigilância será de 3% [12], indicando uma crescente demanda no setor de segurança com o uso da Internet. Um fator importante de contribuição que podemos observar para o crescimento de tráfego de vídeo na Internet, além das melhorias de infraestrutura e aumento da banda, é a difusão de câmeras IP usadas tanto nos sistemas profissionais de videovigilância quanto na forma doméstica. Os dispositivos de uso profissional analógicos foram aos poucos cedendo espaço para os modelos digitais com acesso a Internet, visto que os modelos tradicionais analógicos de vigilância por circuito fechado de TV (CFTV), apesar de funcionarem muito bem em sistemas pequenos, possuem limitações como a falta de acesso remoto, criando uma notória dificuldade de integração com outros sistemas [41].

As câmeras IP podem ser acessadas diretamente pela Internet[1], desde que sejam instaladas em qualquer meio compatível com ou sem fio, e ocasionalmente com alimentação por PoE (*Power over Ethernet*), permitindo alta escalabilidade, ascendendo um olhar em direção a ubiquidade para o qual sistemas IoMT podem ser usados para melhorar a capacidade dos modelos de vigilância e segurança.

1.1 Definição do problema

Em uma grande sala de controle como o Centro de Operações[2] da prefeitura do Rio de Janeiro, aparecem vários monitores com informações de câmeras em tempo real. São aproximadamente 1500 câmeras sendo 800 do próprio centro e mais 700 pertencentes a outras entidades que cedem o acesso. É pouco provável que uma pessoa consiga observar muitos destes monitores simultaneamente sem uma ajuda computacional que detecte eventos de forma automática. Por exemplo, rastrear uma frota de caminhões através de várias câmeras e monitores no percurso de cidades pode tornar o rastreamento um pouco trabalhoso. Estudos mostraram que após apenas 20 minutos de observação e avaliação de telas de monitor, a atenção da maioria dos indivíduos degenerou abaixo dos níveis

[1] Utilizando um provedor de Internet e acessível por algum IP
[2] http://cor.rio

aceitáveis [8] [21].

Uma forma de tratar este problema seria os caminhões utilizarem um sistema de monitoramento geoespacial (coordenadas geográficas) em conjunto com as informações que representem as áreas de cobertura das câmeras por onde os caminhões estivessem trafegando. Detes modo, quando as coordenadas geográficas do caminhão estivessem inseridas em uma área de cobertura de alguma câmera as imagens seriam enviadas para um único monitor, de modo que os vídeos fossem sendo substituídos ou chaveados para as próximas câmeras na medida em que houvessem a transposição de novas áreas de cobertura. O serviço com base em geolocalização e áreas de cobertura de câmeras pode ser utilizado para outras finalidades envolvendo segurança, por exemplo, monitorar pessoas em tempo real. Com as opções de localização geográfica em conjunto com vídeo é proposto um mecanismo de apoio para sistemas de segurança.

Existem recursos de computação visual desenvolvidos para o reconhecimento de pessoas e objetos, onde o rastreamento por vídeo é umas das questões-chave, por exemplo, no processo de localizar um ou vários objetos em movimento ao longo do tempo, sendo muito utilizado nos campos da robótica, controle de tráfego, segurança e vigilância [35][55][25] [47][51][54], mas exigem uma demanda de processamento significativo para analisar as imagens de diversas câmeras.

Atualmente, vivemos em um período de popularização dos serviços baseados em localização (LBS – Location-based Service), no qual os serviços podem ser de qualquer tipo, mas levando em conta a localização geográfica de uma entidade (humana ou não) [12]. Como exemplo destes serviços, temos o monitoramento veicular, visualização de tráfego e análise de qualidade para encontrar melhores rotas, solicitar a entrega de alimentos com base nos locais mais próximos e até mesmo para chamadas de emergência como E911 (Enhanced 911) [6]. O E911 é uma melhoria do sistema de chamada de emergência 911, que habilita o envio de coordenadas de localização, em que na fase I, usa a posição de referência da estação base de transmissão celular e, na fase II, informações mais precisas como latitude e longitude do chamador [14].

Com a popularização do recurso de GPS em telefones móveis ou mesmo em dispositivos rastreadores bem pequenos, pode-se avaliar a possibilidade de utilizar esta tecnologia para viabilizar um processo de monitoramento com baixo custo computacional em conjunto com câmeras de vigilância, proporcionando uma abordagem de monitoramento sem a utilização de computação visual. Contudo, o monitoramento em tempo real por LBS pode fornecer localizações, mas não pode prover informação visual de câmeras próximas

para mostrar o que está acontecendo com a entidade monitorada, portanto a unificação com o sistema de vídeo é um problema que merece atenção.

Como problema a ser explorado, identificou-se que a integração dessas informações (Posição geográfica e Vídeo) poderia melhorar os sistemas de vigilância a partir de uma perspectiva de segurança para a realização de monitoramento de uma entidade e acompanhamento visual simultâneo atravéz de câmeras IP.

1.2 Objetivos

O objetivo geral deste trabalho é propor uma arquitetura georreferenciada (baseada em localização) para seleção de fluxo de vídeo, que é usada para monitorar entidades (pessoas ou não) em um contexto de Internet de Coisas Multimídia (IoMT). Como objetivos específicos, podem ser ressaltados a realização de uma pesquisa bibliográfica sobre o tema para o levantamento do estado da arte, a elaboração de um modelo para a representação de áreas de cobertura de câmeras de monitoramento, a definição de uma abordagem dinâmica para o armazenamento e consulta de informações sobre as câmeras e a implementação de um protótipo para a avaliação da arquitetura proposta.

1.3 Metodologia

A metodologia científica empregada neste trabalho teve como apoio *"Systems development in information systems research"* [42] para pesquisas de sistemas de informação. Foi feita uma revisão de literatura, identificação do problema e formulação de uma proposta que foi validada através de uma prova de conceito.

De acordo com a Figura 1.1 as abordagens multi-metodológicas seguem as etapas a seguir:

- Construção de Teoria: inclui o desenvolvimento de novas idéias e estruturas conceituais, novos métodos ou modelos.
- Experimentação: são as estratégias de pesquisa, como experimentos de laboratório e de campo, bem como simulações computacionais e experimentais.
- Observação: incluindo metodologias de pesquisa, como estudos de caso, estudos de campo e pesquisas por amostragem que são operações de pesquisa não-intrusivas.

- Desenvolvimento de Sistemas: que pode ser dividido nas etapas de concepção de conceitos, construção da arquitetura no sistema, prototipagem, desenvolvimento de produtos e transferência de tecnologia.

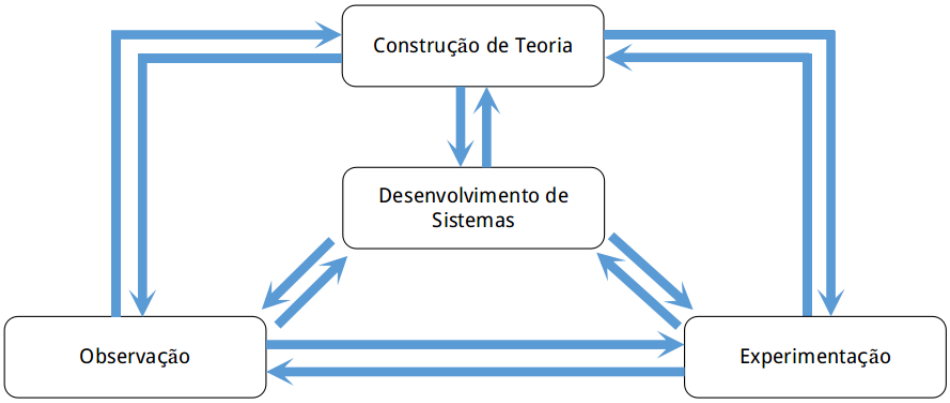


Figura 1.1: Abordagem multi-metodológica para pesquisa de SI. Imagem baseada em [42]

Como parte da construção de teoria foi feito um levantamento e estudo da literatura para identificar os conceitos principais envolvidos. No desenvolvimento de sistemas foi feita a proposta abstrata de uma arquitetura para o monitoramento, e como parte da experimentação e observação foi construído um prototipo com aferição de questões de viabilidade, como por exemplo, o tempo de recebimento de localizações e tempo para a transferência de imagens da câmera.

Como prova de conceito da arquitetura proposta, foi desenvolvido um protótipo de sistema de monitoramento de entidades através de câmeras de vídeo IP, incluindo o desenvolvimento de um servidor de vídeo para a Web. O servidor de vídeo tem a função de primordial de iniciar a transmissão do vídeo da entidade no menor tempo possível, visto que esta pode estar em movimento, a fim de que as suas imagens (da entidade) representem a sua localização real com uma diferença de tempo tolerável.

Pode-se entender como entidade uma pessoa ou objeto que consiga transmitir a sua localização, para que seja identificado o cruzamento destas informações geográficas com diferentes áreas de cobertura, de forma que o monitoramento visual é disparado quando a entidade entrar em uma destas áreas. Portanto, como se trata de um modelo de monitoramento é necessário que as imagens sejam transmitidas com baixa latência para que o elemento monitorado ainda esteja dentro ou próximo do campo de visão da câmera no momento em que as imagens chegem ao observador.

1.4 Organização do texto

No Capítulo 2, são abordados os conceitos básicos sobre os temas principais, como GIS (*Geographic Information Systems*), uso de mapas na web (*Web Mapping*) e Víde Streaming na Internet.

No Capítulo 3, identificam-se os trabalhos relacionados que possuem dentro do tema principal algum item que possa ser avaliado e discutido.

No Capítulo 4, é apresentada a visão geral da arquitetura, a descrição do servidor de monitoramento e seus módulos, as informações sobre a câmera e área de cobertura incluindo um modelo para a representação de áreas de cobertura e uso de metadados, os requisitos para um servidor de vídeo e finalizando com a interface do usuário.

No Capítulo 5, são descritos as formas e técnicas empregadas para o protótipo com um servidor de monitoramento e seus módulos, o servidor de vídeo e interface de usuário para o desenho e cadastro de áreas de cobertura.

No Capítulo 6, é discutida a avaliação realizada do protótipo com um cenário de aplicação e a análise de retultados.

No Capítulo 7 é feita a conclusão deste trabalho com a sugestão de trabalhos futuros.

Capítulo 2

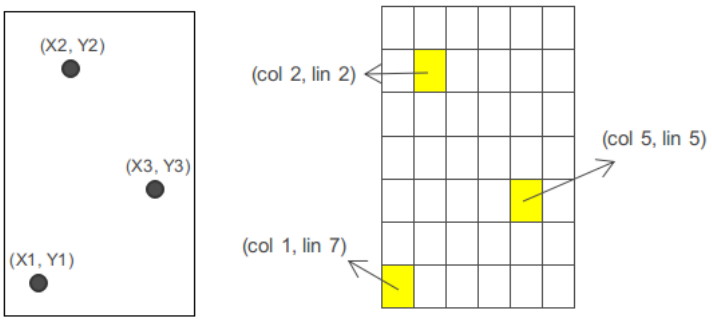
Conceitos Básicos

Neste capítulo, são introduzidas as informações básicas sobre os temas abordados neste trabalho como o *Geographic Information System* (GIS) na Seção 2.1, as características principais a cerca de mapas na Web (*Web Mapping*) na Seção 2.2, bem como uma a verificação de ferramentas de *Geocoding* e edição de mapas nas plataformas OpenStreetMap e Google Maps. A Seção 2.3 aborda streaming de vídeo na Internet, onde são indicados os principais métodos de transmissão de conteúdo multimídia.

2.1 Geographic Information System (GIS)

Geographic Information System (GIS) é um sistema computadorizado para armazenamento, gerenciamento, análise e visualização de dados geoespaciais [6] permitindo a representação do espaço e dos fenômenos que nele ocorrem. A partir da década de 70, tornou-se importante para profissionais interessados em gerenciamento de recursos naturais, transportes, serviços públicos, planejamento urbano e até mesmo para análise de marketing. Após a integração do GIS com a Internet e o sistema global de posicionamento (*Global Positioning system* - GPS), apareceram aplicativos e serviços baseados em localização (*Location-Based Services* - LBS), mapeamento na Web (*Web mapping*), sistemas de navegação para veículos, etc. A tecnologia geoespacial é centrada em GIS e a utiliza para integrar dados de sensores, cartográficos, de GPS e pesquisas para produzir informações geográficas. Um exemplo de utilização de tecnologia geoespacial é abrir o Google Maps, localizar uma referência e adicionar símbolos e comentários ou traçar melhores rotas através de um sistema de navegação veicular.

GIS consiste nos elementos divididos em: dados geoespaciais, bem como sua aquisição, gerenciamento, visualização, exploração e análise. Dados geoespaciais são informa-



(a) Modo Vetorial (b) Modelo Raster

Figura 2.1: Exemplo de modelos de representação de dados geoespaciais na forma Vetorial e Raster.

ções relacionadas à localização geográfica de algum elemento respectivo a algum lugar na superfície da Terra, ou seja, estes dados cobrem a localização de características de cunho espacial [6]. Para que esta representação seja feita, utilizam-se sistemas de coordenadas que são expressadas de forma geográfica em latitude e longitude ou através de coordenadas projetadas. Um exemplo de sistema de coordenadas projetado muito comum é o *Universal Transverse Mercator* (UTM), no qual divide a superfície da Terra entre 84°N e 80°S dentro de 60 zonas.

Os dados geoespaciais podem ser representados como vetor usando pontos, linhas e polígonos ou usando o modelo Raster, o qual usa grades para representar estes mesmos elementos. Na Figura 2.1, é mostrado um exemplo de representação de vetor e Raster. No modelo Raster, um ponto é representado por uma única célula, uma linha através de uma sequência de células vizinhas e de polígonos com uma coleção de células contíguas. Representações em Raster são úteis, por exemplo para a digitalização de elevações e imagens de satélite, e o vetorial para o contorno de vegetações, loteamento de terrenos, etc.

A aquisição de dados envolve a compilação de dados preexistentes e de novos dados, como mapas de satélite, endereços de ruas, ou qualquer procedimento que recolhe amostras do mundo físico e o transforma para o mundo digital. O gerenciamento destes dados serve para manipular todas as informações em algum banco de dados.

A exibição de dados em um GIS combina elementos como corpo do mapa, legenda, escala, etc. Após a reunião dos elementos que compõem um mapa, é possível que ele seja impresso ou exportado através de conversão para arquivos KML (*Keyhole Markup Language*) que podem armazenar polígonos, lugares, descrições textuais, etc. Análise é

o processo de inferir significado aos dados e podem ser realizadas por medições, computação estatística e outras operações. Geralmente as ferramentas para análises específicas não se encontram no GIS e por isso dados são exportados para outros computadores ou programas. Na Tabela 2.1, são apresentados alguns exemplos de softwares GIS.

Tabela 2.1: Exemplos de alguns softwares de GIS

Comercial	**Free ou Open Source**
ArcGIS, AutoCAD Map3D, Bentley Map, GeoMedia, Global Mapper, Manifold System, MapInfo, Maptitude, Smallworld, TerrSet/IDRISI, Cadcorp	GeoDa, GRASS, gvSIG, ILWIS, MapWindow, OpenJump, QGIS, SAGA GIS, uDig

Geralmente os usuários de GIS utilizam características espaciais da superfície da Terra em forma plana com mapas. As localizações no mapa são baseadas em sistemas de coordenadas expressas com X e Y, mas para localizar algo na superfície da Terra baseia-se em coordenadas geográficas expressas em longitude e latitude que são medidas angulares, como visto na Figura 2.2. As medidas angulares podem ser expressadas em graus-minutos-segundos (DMS, *degrees-minutes-seconds*), graus decimais (DD, *decimal degrees*) ou em radianos. A longitude mede o ângulo a leste ou a oeste a partir do meridiano principal ou meridiano de Greenwich indo de 0°até 180°, e a latitude mede o ângulo para norte ou para o sul a partir da linha do Equador com valores de 0°até 90°. As linhas do meridiano principal e do Equador servem como base para o sistema de coordenadas geográficas. Um mapa de projeção é uma ponte entre dois tipos de coordenadas, onde o processo transforma a superfície terrestre em um plano. Uma projeção é uma conversão de um conjunto de dados de coordenadas geográficas para coordenadas projetadas, em outras palavras, é um arranjo pelo qual a superfície curva da Terra é representada como um plano [26], e reprojeção é a conversão de um sistema de coordenadas projetadas para algum outro sistema.

A Terra possui um alargamento ao longo do Equador se comparado aos polos, sendo assim sua forma é aproximada a de um elipsoide. Coordenadas geográficas baseadas em elipsoide são conhecidas como coordenadas geodésicas, que são a base para todos os sistemas de mapeamento. Contudo, existem distorções ao se fazerem projeções de mapa, pois cada projeção preserva certas propriedades espaciais em detrimento de outras. Em geral, a forma, a área e o tamanho de certas características na superfície da esfera ou do elipsoide serão diferentes quando convertidos para a projeção.

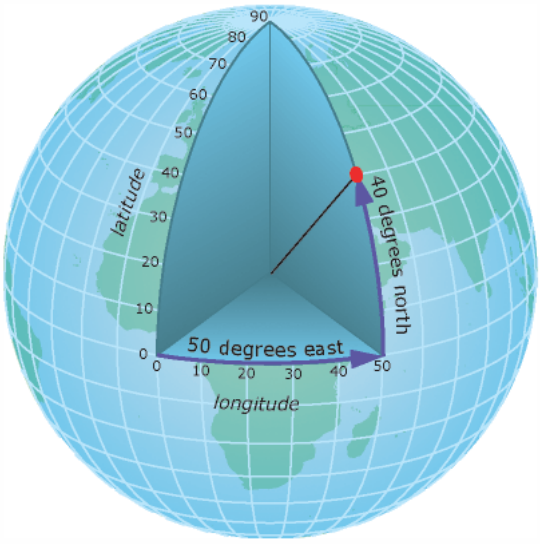


Figura 2.2: Representação de longitude e latitude. Fonte: desktop.arcgis.com

2.2 Web Mapping

Web mapping permite ao cliente de uma página Web poder obter mapas e imagens de um servidor através da plataforma Web, bem como requisitar dados e fazer pesquisas. Apesar deste serviço já ser algo comum no início do ano 2000, apenas após o ano de 2005 se tornou popular com o surgimento do Google Maps e do Google Earth. Este sucesso alavancou outras companhias a usarem serviços comparáveis como Bing Maps, Yahoo! Maps, Apple Maps e HERE [6]. O projeto OpenStreetMap (OSM) foi criado em 2004 [40] e traz um conceito de edição colaborativa, onde os usuários podem editar mapas através de um modelo inspirado por sites como o Wikipedia. Em seguida são discutidas as plataformas de mapas Google Maps e OpenStreetMap.

2.2.1 OpenStreetMap - OSM

Os dados do OpenStreetMap são disponibilizados de forma aberta sob a licença *Data Commons Open Database License* (ODbL)[1], a qual dá a liberdade para copiar, distribuir, transmitir e adaptar dados. Existem vários ambientes para utilizar mapas tanto online como offline, porém, especificamente na Web pode-se listar alguns modos de uso como, por exemplo, no próprio endereço (https://www.openstreetmap.org) ou incorporando mapas em uma página Web por meio de códigos HTML.

Para editar mapas, é necessário utilizar algum editor que pode ter a versão online

[1]https://www.openstreetmap.org/copyright/en

usada diretamente nos navegadores, por exemplo o iD (Programado em JavaScript), Potlatch 2 (Usa tecnologia Flash), a versão offline para Desktop (JOSM, Merkaartor), a versão móvel (Vespucci, StreetComplete, OsmAnd) e multiplataformas (Quantum GIS). O ArcGIS possui plugins para editar dados do OpenStreetMap, contudo, é um software de propriedade da ESRI[2].

São abordadas neste texto apenas as funcionalidades que são úteis na proposta deste trabalho, como a **pesquisa de endereços** e **desenho de geometrias** sobre o mapa.

Pesquisa de endereços: A pesquisa de endereços é parte fundamental, pois, com esta ferramenta é possível para um cliente Web procurar por câmeras em alguma região. Basicamente o endereço requisitado é enviado para algum servidor que faça o Geocoding, que é um termo usado para descrever a conversão de um endereço postal em modo de texto digital para uma coordenada geográfica como latitude e longitude [20]. As pesquisas de endereços são realizadas por ferramentas chamadas *Search engine services*, que utilizam os dados do OpenStreetMap para indicar o local no mapa. Exemplos de plataformas de pesquisa são Nominatim, LocationIQ.org, OpenCage, Photon, GeoCheck entre outros. Para cada uma destas plataformas, foram dados endereços de ruas e avenidas para avaliar a precisão dos locais apontados nos mapas. Na avaliação realizada somente Nominatim e Photon obtiveram êxito para a maioria dos locais, sendo que alguns foram apontados a uma boa distância do local correto.

Geometrias: As geometrias em um mapa na Web são formas empregadas para representar posições ou áreas do mundo real. No OpenStreetMap os componentes básicos são chamados de Elementos, que são divididos por *Nodes*, *Way*, *Relations*, *Tag* e *Common attributes* conforme a Tabela 2.2.

Na versão atual de sua API (V6) é permitido que se faça apenas a edição dos mapas de forma permanente, ou seja, os dados são atualizados para que a comunidade as utilize. Para fins deste trabalho, não é necessário que se faça alteração na base de dados do OSM, mas apenas que se consiga fazer alterações temporárias sobre o mapa, como adicionar marcadores e desenhar polígonos para representar o local de uma câmera e sua área de cobertura. Também é interessante que estes componentes possam ser exportados e importados de uma base de dados própria.

Para incorporar elementos sobre o mapa, existe uma API chamada Overlay que trabalha com vetores e marcadores sobre um *Slippy Map*. Um *Slippy Map* é uma versão de mapa onde é possível dar zoom e arrastar para os lados dinamicamente. A Overlay API

[2]https://www.esri.com/en-us/home

Tabela 2.2: Tipos de elementos do OSM

Elemento	**Descrição**
Node	Representa um ponto específico na superfície da Terra e é definido por um identificador (id) e um par de coordenadas latitude e longitude.
Way	Formado por uma lista ordenada de Nodes usados para representar estradas e rios. Quando o primeiro e o último Node são os mesmos então existe uma área fechada chamada de polígono, que é usado para representar prédios e florestas.
Relation	É uma estrutura de dados usada para registrar relações entre dois ou mais elementos (nodes, ways, relations). Exemplo: Uma rota, restrição de caminhos e multipolígonos. Os tipos de relações (Relations) são definidos por Tags.
Tag	Todos os elementos podem ter Tags, no qual descreve o significado de um elemento em particular (Como um atributo geográfico) e consiste em dois termos: chave e valor (ex:highway=residential) significando que a estrada define local onde moram pessoas.
Common attributes	São atributos associados a Nodes, ways e Relations. Exemplo: iD, user, uid, timestamp, visible, version, changeset. Nesta ordem estes atributos significam o identificador do elemento, nome do último usuário que modificou o objeto, o número de identificação do último usuário que alterou o objeto, o timestamp da modificação do objeto, propriedade pode ser verdadeira (True) ou falsa (False) indicando se o objeto deve ser deletado ou mantido, número da versão indicando quantas vezes houve atualização do objeto, número que indica um conjunto de alterações feitas pelo usuário em um pequeno intervalo de tempo (ex: adição de novos elementos, alteração de Tags, etc).

usa a estrutura de objetos em GeoJSON para montar sua hierarquia de dados interna e isso permite fazer a leitura de um arquivo com muitos componentes (pontos, linhas, e polígonos) para serem carregados de uma só vez no mapa. Apesar destas promessas de funcionalidades, existem poucas documentações para esta API, portanto, esta ferramenta não foi encorajada.

Existem diversos projetos que trabalham com OpenStreetMap que são separados por linhas específicas, sendo que nem todos possuem uma boa documentação. Alguns exemplos de projetos que podem trabalhar de forma dinâmica com mapas na plataforma Web

através de códigos HTML e JavaScript com GeoJSON incluem **Leaflet**, **Mapbox GL JS**,**OpenLayers** indicados nas Figuras 2.3, 2.4, 2.5.

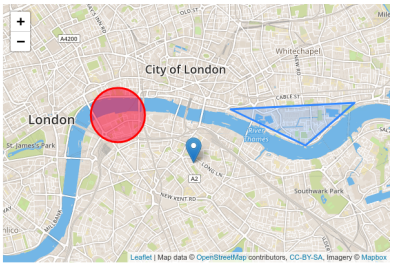


Figura 2.3: Exemplo com Leaflet

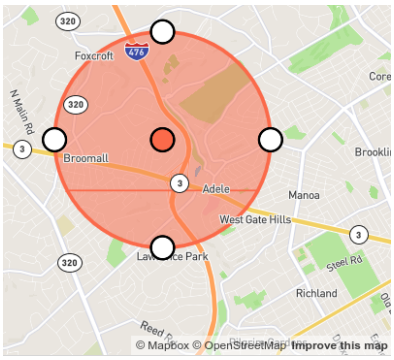


Figura 2.4: Exemplo com Mapbox

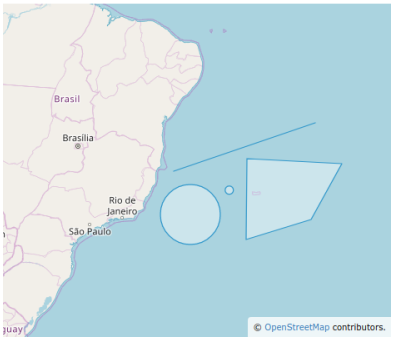


Figura 2.5: Exemplo com Openlayers

2.2.2 Google Maps

Há também opções de ferramentas disponíveis para pesquisa de endereços e desenhos de geometrias (Pontos, linhas e polígonos). O Google Maps disponibiliza mapas estáticos, dinâmicos, imagens de rua (Street View) e visualização em 360°. Na Tabela 2.3, é indicado um conjunto de ferramentas disponíveis para trabalhar com mapas.

Tabela 2.3: Tabela de ferramentas do Google Maps

API e SDK' de Mapas
Maps SDK for Android
Maps JavaScript API
Maps SDK for iOS
Street View API
Maps Static API
URLs do Maps

Os Kits de desenvolvimento de software (SDK) servem para a criação de aplicativos nos sistemas operacionais Android e iOS. Com JavaScript API é permitido adicionar mapas interativos e customizá-los com um conteúdo individual. É possível usar camadas (ex: Dados, Trânsito, etc), estilos, controles, eventos, etc. Na Figura 2.6 é exibido o desenho de figuras geométricas sobre o mapa usando a biblioteca Drawing Tools. No Street View é possível adicionar uma imagem panorâmica estática dentro de uma página Web sem a necessidade de JavaScript, através de uma simples URL. Também é possível usar URLs para fazer pesquisas, descobrir direções e exibir imagens de rua (Street View). Entre as APIs da Tabela 2.3 a que será a base deste trabalho é a *Maps JavaScript API*.

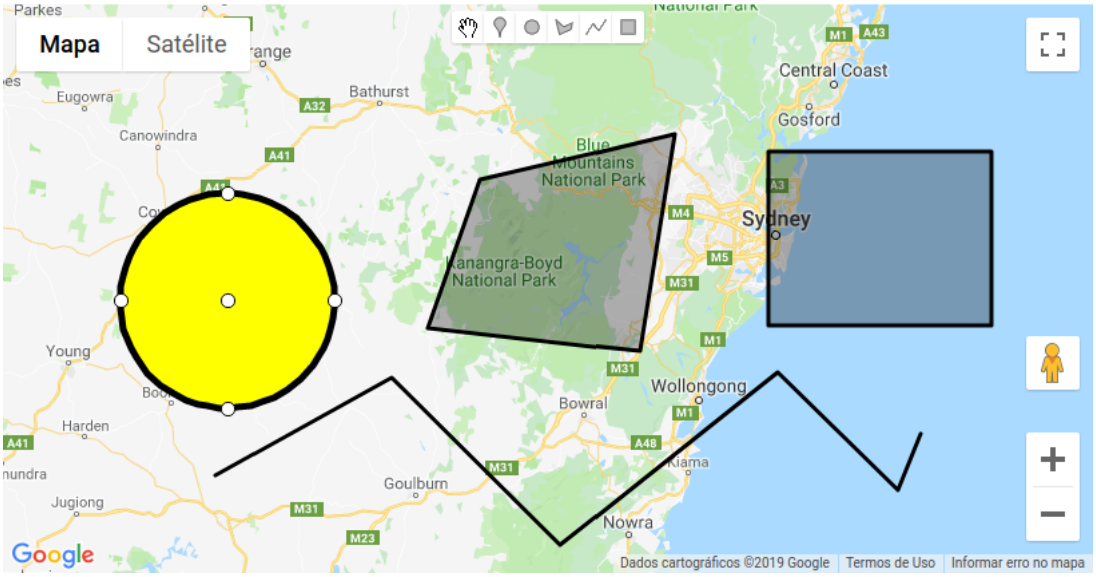


Figura 2.6: Exemplo de figuras geométricas no Google Maps.

Na Tabela 2.4 estão colocadas as APIs relacionadas a mais de 100 milhões[3] de lugares permitindo que sejam usados para pesquisas os números de telefone, endereços, etc.

Tabela 2.4: Tabela de ferramentas do Google Maps

API Places
Places SDK for Android
Places API for Web
Time Zone API
Places SDK for iOS
Geocoding API
Places Library, Maps JavaScript API
Geolocation API

Entre as opções da Tabela 2.4 as utilizadas neste trabalho serão a *Geocoding API* e *Geolocation API*. Com o JavaScript API conseguimos construir geometrias na camada de Dados e exportá-las no formato GeoJSON. Geocoding, como descrito na Seção 2.2.1, significa o processo de conversão de endereços (ex: ruas e avenidas) para coordenadas geográficas de onde estes locais se encontram na superfície do Globo. O contrário também é válido, ou seja, fazendo a pesquisa com as coordenadas geográficas é retornado o endereço da localidade, que é chamado de Geocoding reverso (Reverse Geocoding). Geolocation API é uma alternativa ao GPS quando não está disponível, visto que pode calcular a localização baseando-se em torres de celulares e de roteadores Wifi detectáveis pelos dispositivos móveis.

[3]https://developers.google.com/maps/documentation/

2.2.3 Comparação entre as Plataformas

O ponto forte do OpenStreetMap é a quantidade de ferramentas disponíveis de terceiros que utilizam o seu banco de dados de mapas. Umas são melhores documentadas que as outras, porém, devido a esta extensa família de projetos com funcionalidades muito variadas, torna-se este também o ponto fraco pela grande heterogeneidade de sistemas. Iniciar um projeto com base em uma das propostas de ferramentas e depois identificar problemas pode tornar o trabalho muito extenso e pouco útil. Ao contrário disso, o Google Maps concentra sobre o mesmo mantenedor as APIs com diversas funções úteis e com boa documentação, incluindo vários exemplos de caso. Na Tabela 2.5, foi realizada uma verificação para a contabilização[4] de páginas Web que utilizam os serviços de mapas das plataformas indicadas. Como identificado, o Google Maps ainda é a solução mais utilizada.

Tabela 2.5: Comparativo de utilização de plataformas

Plataforma	**Quantidade de páginas web utilizando a plataforma**
Google Maps	3.839.960
Leaflet	83.668
OpenLayers	12.904
OpenStreetMap	8.965
Bing Maps	7.647

2.3 Vídeo Streaming na Internet

A transmissão de vídeo na Internet é feita por pessoas do mundo inteiro e a visualização sob demanda é impulsionada através de serviços como Netflix, Youtube, Youku (Versão chinesa do Youtube), Hulu, PrimeVideo, entre outros. Além destes, existem os aplicativos como Skype e Google Hangouts, que permitem ligações telefônicas e chamadas de vídeo. Basicamente as transmissões de áudio e vídeo são de três tipos: fluxo de vídeo contínuo armazenado, voz e vídeo interativos e fluxo de vídeo contínuo ao vivo. No fluxo contínuo armazenado a mídia já está gravada, como um filme, programa de TV, um evento esportivo ou algum vídeo gravado e enviado pelo usuário como ocorre no Youtube. Após a colocação destes arquivos em um servidor ou CDN (*Content Delivery Network*), outros usuários poderão acessá-los sob demanda, como o exemplo mostrado na Figura 2.7. Neste

[4]similartech.com

caso, o fluxo contínuo acontece quando a reprodução da mídia no usuário é feita logo após o recebimento de parte do conteúdo, que é armazenado em memória por frações de segundo até o momento de ser reproduzido enquanto novas partes do vídeo continuam sendo recebidas.

Figura 2.7: Exemplo de fluxo de vídeo contínuo armazenado

Voz e vídeo interativos são as aplicações de chamadas de voz, ou voz e vídeo em tempo real na Internet. Para o caso de ser somente a voz (VoIP) funcionaria como se fosse um serviço telefônico tradicional do ponto de vista do usuário. Para chamadas de vídeo, ocorre de maneira semelhante, mas adicionalmente são transmitidas as vozes dos usuários. Os sistemas interativos, em sua maioria permitem videoconferências (ex: Skype). Na imagem da Figura 2.8 é mostrado um exemplo de video conferência por Skype.

Figura 2.8: Exemplo de voz e vídeo interativos

Na transmissão de fluxo contínuo ao vivo, o usuário recebe o conteúdo pela Internet na medida em que o vídeo é criado e transmitido até o receptor final, semelhantemente ao modelo de transmissão de TV. Embora a distribuição para muitos receptores fosse mais eficiente utilizando técnicas IP para transmissão a grupos, atualmente os envios de vídeo são comumente realizados por aplicações P2P, por CDNs ou com vários fluxos individuais. Como o evento é ao vivo existe o problema do atraso, embora atrasos de até 10 segundos possam ser tolerados para a transmissão ao vivo [30]. Na Figura 2.9 é mostrado um exemplo de transmissão de fluxo contínuo ao vivo.

Figura 2.9: Exemplo de transmissão de fluxo contínuo ao vivo

Para o transporte de áudio e vídeo em tempo real, a IETF (*Internet Engineering Task Force*) designou o protocolo RTP (*Real-Time Transport Protocol*), que provê funções da rede de transporte para aplicações de tempo real. O RTP pode ser usado em conjunto com o protocolo de controle RTCP (*Real-Time Transport Control Protocol*). Redes gerenciadas estão deixando lugar para as redes de conteúdo (CDN) onde muitas das quais não suportam streaming RTP [46]. Também é requerido para o RTP um servidor para gerenciar sessões para cada cliente tornando a escalabilidade algo a ser bem avaliado. O RTP é tipicamente utilizado pelas aplicações sobre o UDP, mas o uso de UDP para fluxo contínuo possui algumas limitações básicas, como a variação da largura de banda entre o servidor e o cliente, no qual uma transmissão UDP com taxa constante pode deteriorar a reprodução multimídia. Todos os protocolos de transporte precisam abordar o controle de congestionamento de alguma forma e o RTP não é uma exceção [17]. No RTP o meio de fazer o controle de congestionamento é diferente dos demais como o TCP, sendo que o risco de congestionamento é reduzido pela inelasticidade (fluxo gerado a uma taxa fixa ou controlada), pois o fluxo RTP não é aumentado a fim de consumir toda a banda disponível como o TCP pode fazer. Outra limitação é que, em geral o tráfego UDP é bloqueado em firewalls, ao contrário dos serviços HTTP que têm livre passagem na maioria dos casos.

Com o aumento da largura de banda da Internet e o crescimento da rede mundial de computadores (*World Wide Web*) o custo de entrega de áudio ou vídeo em pequenos pacotes diminuiu. Conteúdos multimídia puderam ser entregues eficientemente por segmentos maiores usando HTTP e desenvolvimentos para *streaming* de vídeo se tornaram populares como *HTTP Live Streaming* (HLS[5]) da Apple, *Smooth Streaming* [6] da Microsoft e *HTTP Dynamic Streaming* [7] da Adobe. Cada implementação usa seus próprios padrões e formatos e, para que o cliente pudesse receber conteúdos multimídia independentemente do servidor, seria necessária a definição de um padrão único. Por uma demanda da indústria e de perspectivas de mercado o grupo MPEG[8] juntamente com especialistas de outros grupos desenvolveram o MPEG-DASH[9] (*MPEG-Dynamic Adaptive Streaming over HTTP*). Baseado em HTTP, o DASH é capaz de transmitir diferentes versões de um vídeo gravado a taxas diferentes com vários níveis de qualidade para se adaptar à largura de banda do cliente. As informações destes conjuntos de arquivos de qualidades diferentes são armazenadas em um arquivo MPD (*Media Presentation Description*) que é enviado ao cliente contendo as URLs e informações das mídias disponíveis.

[5]https://tools.ietf.org/html/rfc8216
[6]https://www.iis.net/downloads/microsoft/smooth-streaming
[7]https://www.adobe.com/products/hds-dynamic-streaming.html
[8]https://mpeg.chiariglione.org
[9]https://mpeg.chiariglione.org/standards/mpeg-dash

O HLS é um serviço popular para a transmissão de áudio e vídeo de modo contínuo na Internet permitindo ao cliente adequar a taxa de bits da mídia para as condições da sua rede tentando deste modo evitar perda de qualidade. O servidor HLS usa como padrão o protocolo HTTP para enviar uma sequência de arquivos MPEG-2 TS (*Transport Stream*) com encapsulamento de vídeos codificados com H.264 ou arquivos MPEG-4 fragmentados [3]. O primeiro arquivo a ser lido no cliente é um índice contendo uma lista ordenada (*playlist*) com uma sequência de nomes de segmentos de mídia ou arquivos, que podem ou não conter variação de fluxos com taxas de bits deferentes. HLS é nativamente aceito no navegador Safari, mas para outros navegadores é preciso utilizar um arquivo JavaScript[10] ou algum plugin HLS. A Figura 2.10 ilustra um exemplo de funcionamento.

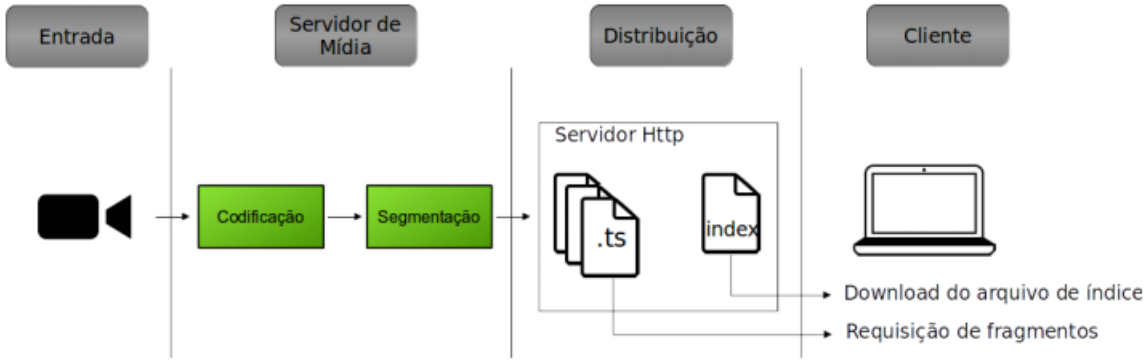


Figura 2.10: Visão global do *HTTP Live Streaming* (HLS)

Em geral as implementações com streaming de vídeo contínuo pela Internet utilizam players de terceiros (não nativos em HTML) para intermediar o recebimento destes fluxos de servidores de streaming de vídeo. Plugins como o Flash Player, que atualmente pertence a Adobe, podem reproduzir conteúdo multimídia contido em arquivos SWF ("swiff"). O Flash Player também pode reproduzir streaming de áudio e vídeo através do formato F4V (MPEG-4 Part 12)[11] e FLV. Atualmente o Youtube permite incorporar seu player em HTML5 em lugar do Flash Player, usando a API IFrame[12].

WebRTC[13], é uma plataforma livre suportada pela Google, Mozilla e Opera permitindo aplicações de tempo real em dispositivos móveis e na Web. Praticamente todos os navegadores modernos possuem suporte ao WebRTC facilitando chamadas de áudio e vídeo sem a necessidade de um plugin. Sua API e protocolos estão sendo desenvolvidos em conjunto com a W3C e IETF. Para ajudar na estabilização da rede, o projeto utiliza RTP-over-TCP (Pacotes RTP e RTCP sobre transporte orientado a conexão), e

[10]https://github.com/video-dev/hls.js
[11]https://www.adobe.com/devnet/f4v.html
[12]https://developers.google.com/youtube/
[13]https://webrtc.org

também utiliza mecanismos para estabilizar conexões através de vários tipos de rede como STUN/TURN/ICE.

É bastante comum câmeras IP trabalharem com RTSP (*Real Time Streaming Protocol*)[14], que é um protocolo da camada de aplicação para o controle da entrega de dados em tempo real para fluxos de áudio e vídeo. Como o padrão HTML5 não possui suporte nativo na tag <VIDEO> para RTSP, alguns plugins são necessários para reproduzir os vídeos, ou utilizar algum método de transcodificação como mostrado na Seção 5.2.

[14]https://www.ietf.org/rfc/rfc2326.txt

Capítulo 3

Trabalhos Relacionados

Neste capítulo são comentados alguns trabalhos que trazem referência aos temas abordados nesta dissertação. Na Seção 3.1 são vistos trabalhos que utilizam serviços baseados em localização (LBS) no contexto de monitoramento. Na Seção 3.2 é abordada uma infraestrutura para a localização de sensores ao redor do mundo pela Internet. Na Seção 3.3 é descrito um sistema que utiliza câmeras distribuídas e a realização de mapeamento de coordenadas geográficas de usuários por meio de calibragem de câmeras. Na Seção 3.4 é abordada uma plataforma de integração de dispositivos do mundo real para a Web usando arquitetura REST (*Representational State Transfer*). Na Seção 3.5 é apresentado um sistema de vigilância que pode controlar diversos sensores incluindo a localização de objetos a partir de câmeras. Na Seção 3.6 é descrito um framework de videovigilância com base nos princípios de IPTV. Na Seção 3.7 é apresentado um sistema de reconhecimento visual de placas de automóveis, seus desafios e problemas relacionados. Na Seção 3.8 são descritas as principais funções dos centros de comando e controle, com um pequeno resumo de algumas tecnologias empregadas para o processamento de imagens das câmeras de vigilância por métodos de computação visual.

3.1 Serviços baseados em localização (LBS) no contexto de monitoramento

Há uma tendência crescente para serviços baseados em localização, e como exemplo de tecnologia bastante empregada tem-se o GPS que pode realizar rastreamento, registrar padrões de objetos em movimento como os de animais, mobilidade humana, estudos de tráfego urbano, e nas últimas décadas tem sido, por exemplo, usado frequentemente para rastrear e monitorar veículos de transporte de carga.

A riqueza destes serviços cria uma expectativa e excitação para a maioria dos usuários para a sua utilização, o que pode acabar negligenciando a importância da precisão de localização dependendo do tipo de dispositivo usado.

No caso do GPS a precisão dos dados depende de muitos fatores como a qualidade do equipamento de recepção, da posição do satélite, das características da paisagem circundante que podem bloquear os sinais que causam erros de *multipath*, etc. [43]. Mesmo assim, tecnologias para localização em tempo real podem usar algoritmos para melhorar a precisão como em [9] [27] [45].

Em [39] um sistema é proposto para monitoramento de veículos através de um microcontrolador e uma aplicação móvel para localização de veículos no google maps. Dados de coordenadas são passados para o usuário através de uma conexão GSM via SMS. Nenhum método é relatado para verificar o atraso da mensagem por SMS, no entanto, no dia a dia, podemos ver flutuações significativas no tempo de recebimento. Embora a recepção de coordenadas de satélite seja considerada em tempo real, toda a rota da informação e o tempo relacionado precisam ser considerados até que cheguem ao usuário final. A mesma coisa acontece em [36] [44]. No entanto, deve-se notar que nem todos os sistemas que utilizam o GPS exigem a entrega imediata de dados, como por exemplo o de proteção antirroubo [13], que faz a coleta da impressão digital do motorista.

Em [11] os autores propõem um sistema de monitoramento inteligente para e-bikes. É uma plataforma para coletar, analisar e compartilhar dados sobre a frota de bicicletas suportada por um sistema de bateria. A cada 25 segundos um telefone pareado é ativado para coletar e enviar dados em um formato compactado. O telefone grava a latitude, longitude e altitude uma vez por segundo. A informação é transmitida através da rede de celular (3G). Embora o envio de dados para este monitoramento não seja realizado o tempo todo, ele fornece uma boa maneira para a entrega de dados via Internet.

3.2 Arquitetura IrisNet

Pensando em um modo de conectar sensores ao redor do mundo pela Internet, mais especificamente utilizando câmeras web, Gibbons et al. [19] desenvolveram uma arquitetura chamada IrisNet (*Internet-scale Resource-Intensive Sensor Network Services*) provendo infraestrutura de *software* para localizar sensores globalmente distribuídos e gerenciar os dados coletados pelos mesmos. Os dados são armazenados próximos aos nós geradores e distribuídos de forma hierárquica e geográfica. A arquitetura proposta pelos autores pos-

sui uma complexidade no armazenamento e distribuição de dados, que pode inviabilizar sua implementação. Também não faz um tratamento dos serviços baseados nas áreas de cobertura das câmeras, ou seja, um usuário ou o criador de serviços precisam trabalhar com filtros de procura ou abrir todas as câmeras para conseguir fazer esta identificação visualmente. Com relação ao armazenamento e acessos concorrentes, atualmente poderiam ficar a cargo de servidores de conteúdo ou de banco de dados distribuídos.

O trabalho proposto nestsa dissertação faz o armazenamento dos metadados de câmeras e provê mecanismos para encontrá-las através de seus endereços na Internet. A busca de metadados se dá pelo monitoramento de entidades, mas há também a possibilidade de usar qualquer navegador web colocando no endereço uma posição geográfica e um raio de alcance (ex: http://elementos.com.br/lat/long/raio). O resultado dessa pesquisa retornaria os metadados dos elementos que fossem encontrados.

3.3 Abordagem de Monitoramento Ubíquo Georreferenciado

Lial et al. [34] propuseram um sistema chamado GODTA (*GPS-based Object Detection and Tracking Approach*), que faz a integração de câmeras distribuídas com a identificação visual de objetos equipados com sensor de GPS (*Global Positioning System*) através de um mapeamento de coordenadas, onde após o usuário estar visível o sistema tenta escrever na tela um identificador para este usuário. O mapeamento é realizado a partir do registro das coordenadas de cinco posições de acordo com método de Lenz e Tsai [33]. Quando o mapeamento é estabilizado, dois parâmetros são computados: parâmetros intrínsecos (distância focal) e extrínsecos (posição e orientação da câmera baseados no plano real em três dimensões). A partir disso, os pontos de coordenadas de GPS podem ser representados em coordenadas de imagem, ou seja, dentre vários indivíduos que são vistos é feita uma identificação na imagem daquele que estiver enviando a sua localização. O método de transformação de coordenadas com origem tridimensional (superfície da Terra) para coordenadas cartesianas em um mapa 2D é chamado TM2 (*2-degree Transverse Mercator*), e em seguida estas coordenadas sofrem uma nova transformação para coordenadas de imagem utilizando os parâmetros adquiridos na fase de calibragem da câmera. Assim é possível fazer a identificação visual do objeto que estiver portando um dispositivo móvel com GPS. Embora a calibragem de uma câmera seja um esforço único, ainda é um fardo quando há um grande número de câmeras.

Neste trabalho, a localização do usuário não será realizada por processos de computação visual que normalmente usam contornos para identificação na imagem onde foi detectado um objeto. A identificação será realizada apenas com a sua localização geográfica no mapa e visualização em tempo real com as câmeras próximas.

3.4 Integração de Dispositivos Baseada em Serviços Web

A falta de um protocolo de comunicação interoperável entre os objetos reais dificulta a materialização e escalabilidade da Internet das Coisas, e como consequência, a engenharia de software deve propor infraestruturas inovadoras que satisfaçam os requisitos relacionados à natureza da IoT, que são entre outros: heterogeneidade (ex: diferentes objetos, sensores, protocolos e aplicações), dinamicidade (ex: chegada e partida de objetos e sensores) e evolução (ex: suporte para novos protocolos, sensores)[18]. Com o aumento de dispositivos embarcados que possam se conectar a Internet, o uso de protocolos e padrões web pode colaborar com as interações entre estes objetos. Guinard [22] aborda a possibilidade de integração de dispositivos do mundo real com a web, aplicando princípios da arquitetura REST para redes de sensores sem fio e dispositivos inteligentes. Usando a Web como uma plataforma de aplicação, os recursos REST podem ser identificados inequivocamente através de uma URI e sobre estes recursos podem ser aplicadas operações HTTP (*GET, POST, PUT, DELETE*). Uma questão que foi levantada é a dificuldade da identificação dos dispositivos com que se quer interagir, e mesmo que todos os dispositivos ofereçam serviços em páginas web, a forma de encontrá-los não é a mesma do processo textual de procura como em um site de busca.

Os métodos de identificação e localização sugeridos neste trabalho por meio de metadados de câmeras colaboram no contexto da pesquisa de elementos, que apesar de ser focado em câmeras pode ser abstraído para qualquer tipo de recurso, ou serviços. Uma interface de usuário foi desenvolvido para fazer o acompanhamento de entidades, mas também possui função de cadastro e pesquisas de câmeras por endereços textuais. Operações HTTP (*GET*) podem ser realizadas independente da interface do usuário (Página Web) para a obtenção de elementos, possibilitando que qualquer equipamento que use o protocolo HTTP realize pesquisas.

3.5 Sistema de Vigilância Baseado em Contexto

No sistema de vigilância proposto em [38], utilizam-se diversos sensores para detectar atividades humanas suspeitas, obstrução das lentes das câmeras, incluindo a detecção de tamanho de objetos e sua localização. Para determinar a localização ou distância do objeto até a lente da câmera, é necessário primeiro determinar o tamanho do objeto e dispor de outras informações como a altura da câmera, ângulo de inclinação, entre outros dados para conseguir ter uma boa precisão do posicionamento do objeto.

Na proposta desta dissertação, câmeras de terceiros podem ser usadas, sendo a única exigência o endereço IP para que as câmeras possam ser acessadas publicamente, por isso, fica impraticável checar todos os dados indicados para obter algum tipo de localização de objetos, como é feito em [38].

3.6 IPTV em um contexto de Vigilância por Vídeo

Hassan et al. [24] propõem um framework de videovigilância na nuvem baseado nos princípios tradicionais de transmissão de vídeo da arquitetura de IPTV. É proposta uma solução de IPTV utilizando os recursos de infraestrutura em nuvem e os benefícios da tecnologia de IPTV para entregar conteúdo de vídeovigilância para diferentes dispositivos clientes. No caso de sistemas de CFTV o conteúdo de vídeo é armazenado em servidores centralizados que podem se vistos por diversos clientes em uma rede com ou sem fio. Já na terceira geração de sistemas digitais de segurança [50] são geralmente equipados com câmeras IP distribuídas, o que habilita os usuários a acessar conteúdo de vídeo sobre a internet em tempo real. IPTV emergiu nos últimos anos como uma técnica de sucesso para distribuição em tempo real como uma transmissão ao vivo de TV e vídeo sobre demanda baseada em redes IP com ou sem fio com suporte para QoS de experiência. Os autores deste artigo aproveitam a vantagem de distribuição de conteúdo e gerenciamento da arquitetura de IPTV pra aplicá-la em um domínio de vídeo vigilância. Foi criada uma interface de cliente para conectar elementos individuais ao sistema como cameras fixas, câmeras IP e dispositivos móveis.

Um dos principais problemas relatados pelos autores são considerar uma grande flexibilidade para adaptar um sistema de videovigilancia para equipamentos heterogeneos e, a necessidade de mitigar a saturação da largura de banda de rede quando usada por muitos clientes ou a da capacidade de *stream* da câmera por acessos simultâneos de vídeo

ao vivo.

O servidor de vídeo desenvolvido para o protótipo deste trabalho tem bastante flexibilidade e pode acessar qualquer câmera independente do fabricante, desde que possa ser acessada pela Internet. A largura de banda foi levada em consideração e realizado testes de transmissão de vídeo com diferentes codificações para facilitar a escalabilidade do sistema.

3.7 Computação visual para o reconhecimento de placas

Chen et al. [8] apresentam um novo sistema chamado *Snake Eyes* que agrupa a tecnologia de reconhecimento de placas de automóveis e a tecnologia de computação em nuvem afim de realizar a análise de uma grande quantidade de dados para fazer a deteção e rastreamento de um veículo especificado em uma cidade.

Muitos sistemas de vigilância inteligentes empregam computação visual e técnicas de reconhecimento de padrões para análise inteligente de vídeo e alertas de eventos em tempo real como detecção de objetos abandonados e *TripWire* (Recurso utilizado para detectar movimento e a direção de algo através de linhas desenhadas na imagem da câmera). Entre eles também se encontra o reconhecimento de placas de automóveis (Automatic License Plate Recognition - ALPR) [7] [2].

Os autores reconhecem que sistemas de vigilância por vídeo geram uma grande quantidade de dados e acreditam que o problema de análise de dados correspondente não são tratados apropriadamente, onde na opinião do autor a próxima geração de sistemas inteligentes de vigilância deverão ser combinados entre análise inteligente de vídeo e computação de alta performance na nuvem.

Na arquitetura proposta pelos autores existe uma infraestrutura de vigilância digital que consiste de câmeras de vídeo, DVR (Digital Video Recorder) e NVR (Network Video Recorder), etc. Esta infraestrutura é responsavel por capturar imagens e gravar temporariamente em um *storage*. Esta infraestutura é distribuída espacialemte mas o servidor de aplicação utilizado forma um data center virtual centralizado.

A análise de vídeo é realizada após os servidores de análise perceberem novas demandas, e após as tarefas serem enfileiradas é realizada a comunicação com os DVRs para a requisição de arquivos de vídeos. Estes videos são armazenados no data center virutal

junto com os *frames* das placas reconhecidas.

Problemas levantados: algumas câmeras de vigilância estão fora de foco, resultando em imagens borradas, ou a resolução dos vídeos gravados não são bons o sufuciente para realizar o reconhecimento de placas. Ocasionalmnete também existe uma indisponibilidade temporária do *link* de comunicação. Usualmente o número de câmeras que podem ser utilizadas são menores do que o montante total. Um conjunto de configurações utilizadas em uma câmera usualmente não serão adequadas todas as outras câmeras, sendo necessária uma calibração apropriada para alcançar uma boa performance. Outro problema levantado foi a necessidade de aumento da largura de banda de rede.

Neste trabalho, optou-se em usar uma arquitetura para consumo baixo de recursos de processamento através do uso de um sistema de localização de coordenadas. Como observado nos problemas levantados no parágrafo anterior existem uma série de desafios de um sistema de detecção por imagens que podem ser atenuados com a ajuda de um sistema de localização em conjunto com as imagens das câmeras como indicado no Capítulo 4. Além disso, o custo de manutenção de uma infraestrutura particular de câmeras pode ser mais baixo com a agregação de novos equipamentos por meio de compartilhamento de câmeras IP em um sistema para cadastro de equipamentos, que também será abordado neste trabalho.

3.8 Centros de Comando e Controle

Os centros de comando e controle (CCC) no contexto de sistemas de videovigilância possuem uma combinação de disciplinas para o seu funcionamento como processamento de sinais, telecomunicações, gerenciamento etc. A habilidade de reconhecer objetos e humanos, descrever suas ações e interações a partir de informações adquiridas por sensores é essencial para a automatização de vigilância visual.

Sistemas que utilizam inteligência visual lidam com monitoramento em tempo real de objetos persistentes ou transitórios dentro de um ambiente específico. O objetivo destes sistemas é prover de forma automática a interpretação de cenários para entender e prever ações e interações entre os objetos observados. Os estágios principais de processamento de um sistema inteligente são: detecção e reconhecimento de objetos em movimento, rastreamento e análise comportamental [50]. Estes estágios envolvem tópicos de visão de máquina, análise de padrões, inteligência artificial e gestão de dados.

Os sistemas de videovigilância evoluíram tecnologicamente a partir de sistemas ana-

lógicos de CFTV (Circuito Fechado de TV). Estes sistemas consistem em um número de câmeras localizadas em locais remotos e conectadas em um conjunto de monitores, usualmente em um lugar chamado de sala de controle. Na primeira geração, estes sistemas estavam bem maduros tecnologicamente e possuíam boa performance para algumas situações, mas tinham uma dificuldade para a distribuição de imagens e de armazenamento. Atualmente as gravações são de forma digital. Na segunda geração, tem-se a combinação de computação visual com os sistemas de CFTV, com a vantagem de aumentar a eficiência destes sistemas. São nacessários algoritmos robustos para a detecção e rastreamento a fim de realizar análises comportamentais. Nesta geração, as pesquisas se concentram em algoritmos de computação visual para tempo real, aprendizagem automática de variabilidade de cena e padrões de comportamento e a ligação ou ponte entre a análise estatística de uma cena e sua interpretação para uma linguagem natural. Na terceira geração, identificam-se técnicas de sistemas automatizados para uma grande área de vigilância, com informações mais precisas como resultado da combinação de diferentes tipos de sensores, mas enfrentando problemas com o desenho de metodologias, plataformas móveis, integração e comunicação para a distribuição de informações, etc. Pesquisas como técnicas de vigilância com múltiplas câmeras, fusão de dados, etc., fazem parte do escopo desta geração.

Para a detecção de pedestres, vários conjuntos de dados públicos de pedestres foram coletados ao longo dos anos como: INRIA, ETH, TUD-Bruxelas, Daimler, Caltech-USA e KITTI são os mais usados [5]. Todos eles têm diferentes características, fraquezas e pontos fortes. Caltech-USA e KITTI são os *benchmarks* predominantes para a detecção de pedestres. Ambos são comparativamente grandes e desafiadores. O Caltech-USA destaca-se pelo grande número de métodos que foram avaliados e os métodos treinados por este método têm um desempenho sistemático melhor do que métodos a partir do INRIA.

Um objeto desacompanhado de seu dono por muito tempo em local público é considerado um objeto abandonado. A identificação de um objeto abandonado em tempo real pode impedir o ataque terrorista através de um sistema automatizado de vigilância por vídeo. A maior parte dos trabalhos propostos para a detecção por vídeos de vigilância de objetos abandonados são capturados por câmeras estáticas. Por exemplo, Lavee et al. [31] introduziram uma estrutura para analisar um vídeo para detecção de eventos suspeitos para reduzir a tarefa exigente de ordenar manualmente horas de vídeo de vigilância sequencialmente para determinar se ocorreu uma atividade suspeita. Ferrando et al. [15] apresentaram uma técnica de classificação de objetos multinível para classificar objetos

roubados e objetos abandonados. Szwoch [48] propôs recentemente uma nova abordagem para detectar objetos estacionários de vídeos de vigilância, onde neste caso, o autor testou a estabilidade de pixels e, em seguida, clusters de pixels com cores e brilho estáveis extraídos da imagem.

3.8.1 Discussão

Tabela 3.1: Comparativos

Trabalho relacionado	**Usa GPS**	**Uso de Visão Computacional**	**Seleção automática da câmera**	**Tipo de acesso**	**Consumo de Banda**	**Custo de Processamento**
Serviços baseados em localização (LBS)	sim	não	não	SMS e 3G	não informado	baixo
Arquitetura IrisNet	não	não	não	Internet	não informado	baixo
Abordagem de Monitoramento Ubíquo Georrefe-renciado	sim	sim	sim	privado	não informado	mediano
Integração de Dispositivos Baseada em Serviços Web	não	não	não	Internet	não informado	baixo
Sistema de Vigilância Baseado em Contexto	sim	sim	sim	privado	não informado	alto
IPTV em um contexto de Vigilância por Vídeo	não	não	não	Internet	alto	alto
Computação visual para o reconhecimento de placas de automóveis	não	sim	não	não informado	não informado	alto
Centros de Comando e Controle	não	sim	sim	privado	alto	alto

Na Tabela 3.1 são verificados alguns aspectos relevantes sobre um conjunto de trabalhos que foram analisados para esta dissertação. Os principais itens comparados foram a utilização de GPS, o uso de visão computacional, se existe ou não a seleção automática de câmeras, o tipo de acesso utilizado para a entrega de informações, o consumo de banda utilizado para a realização prática da proposta e o custo de processamento.

Os trabalhos relacionados que utilizam com mais ênfase o GPS, como em *Serviços baseados em localização (LBS)*, possuem um custo de processamento baixo se comparado com os trabalhos com base em visão computacional, exceto em *Sistema de Vigilância Baseado em Contexto* em que se utiliza GPS para sensores mas a maior parte do processamento é para o tratamento de imagens, da mesma forma que em *Abordagem de Monitoramento Ubíquo Georreferenciado.* A seleção automática da câmera é realizada em trabalhos que utilizam computação visual, de modo que é necessário um custo de Hardware mais elevado para processamento. O consumo de banda para a maioria dos

trabalhos não foram informados e para os que deram alguma informação era necessário que houvesse mais disponibilidade de banda tanto para os que usaram a rede da Internet ou uma rede privada.

Este trabalho demonstra a utilização de GPS com a seleção automática de câmeras sem a necessidade de computação visual, obtendo assim um baixo custo de processamento. Também são analisados os resultados de testes para a transmissão de vídeo na Internet, pois como mostrado na Tabela 3.1 estes dados não foram informados. Como parâmetros de teste tem-se o consumo de banda e atraso na entrega das imagens, visto que se trata de um monitoramento, no qual as imagens precisam ser entregues rapidamente para o acompanhamento em tempo real do objeto monitorado.

Capítulo 4

Arquitetura Proposta

Este trabalho tem por objetivo propor uma arquitetura georreferenciada para seleção de fluxos de vídeo, mas que também possa ser usada para ampliação dos métodos tradicionais de monitoramento, como os serviços de videovigilância e os de localização auxiliados por GPS. Para isso, foi idealizada uma arquitetura para o monitoramento em tempo real apresentada na visão geral da Seção 4.1. Na Seção 4.2 será apresentada a arquitetura para o Servidor de Monitoramento e com as explicações dos Módulos de Registro, Consulta e Monitoramento. Na Seção 4.3 serão abordados a definicição e as informações da câmera e área de cobertura. Na Seção 4.4 será tratado um modelo para o Servidor de Vídeo e na Seção 4.5 será apresentada a Interface do Usuário.

4.1 Visão Geral da Arquitetura

A Figura 4.1 ilustra os elementos que compõem a arquitetura proposta para o monitoramento geográfico e visual de entidades. Esta arquitetura é composta pelas Entidades que enviam suas coordenadas geográficas para o Servidor de Monitormamento que possui os módulos de Monitoramento, Consulta e Registro de metadados de câmeras. Imediatamente após as localizações serem recebidas, são realizadas consultas em um banco de dados de metadados das câmeras para determinar a presença ou não de câmeras na região. As informações de localização da entidade e resposta da consulta, são encaminhadas do Servidor de Monitoramento para uma Interface de usuário Web no cliente para o acompanhamento das entidades em um mapa. O Servidor Web mantém esta Interface, que é uma página Web e possui a universalidade de exibição em praticamente todos os dispositivos com algum navegador moderno e com acesso a Internet. As imagens transmitidas pelas câmeras passam por um Servidor de Vídeo, que deve funcionar como um agregador

de várias câmeras e responsável por transcodificar os vídeos para serem apresentados na Interface do Usuário no Cliente. O vídeo é iniciado quando uma entidade estiver dentro da área de cobertura de uma câmera, ou seja, dentro do alcance visual e interrompida quando a entidade estiver fora do seu alcance visual.

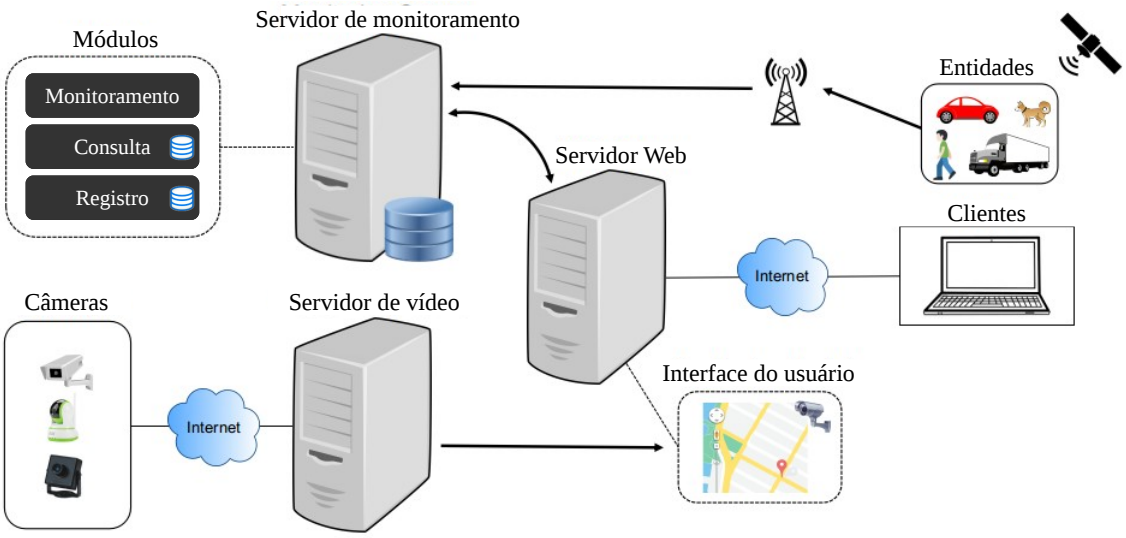


Figura 4.1: Arquitetura de monitoramento

4.2 Servidor de Monitoramento

Na Figura 4.2 é ilustrado em etapas o modo de funcionamento do servidor de monitoramento. Na etapa (1) as coordenadas das entidades são enviadas para o módulo de monitoramento. É necessário que todas as entidades monitoradas utilizem algum equipamento com acesso a Internet e sensor de localização por GPS para enviar as mudanças do posicionamento para o módulo de monitoramento. Na etapa (2) o módulo de monitoramento realiza uma consulta de metadados no módulo de consulta usando como parâmetro o local geográfico enviado pela entidade. Caso a consulta retorne algum resultado, os metadados encontrados são encaminhadas para o módulo de monitoramento e transmitidos para a interface do usuário na etapa (3). Tais metadados incluem, por exemplo, a URI para acessar o dispositivo físico diretamente ou por algum servidor de *streaming* de vídeo, a localização da câmera, o nome do proprietário, etc. As coordenadas de localização da entidade que foram utilizadas na consulta também devem ser encaminhadas para a interface do usuário para ser feito o acompanhamento das entidades no mapa. Para os casos onde não existe cobertura de câmera no local o elemento retornado é nulo, mas ainda sim são as coordenadas são enviadas na etapa (3). Na etapa (4) a interface do usuário de posse dos metadados da câmera encontrada e da URI para acessar a câmera realiza

a apresentação do vídeo com a imagem da entidade dentro da área de cobertura, realizando então a unificação dos sistemas de geolocalização com vídeo. As etapas (a) e (b) representam a consulta e registro de metadados pela interface do usuário. O módulo de consulta também permite que pesquisas externas sejam realizadas através de chamadas HTTP/GET.

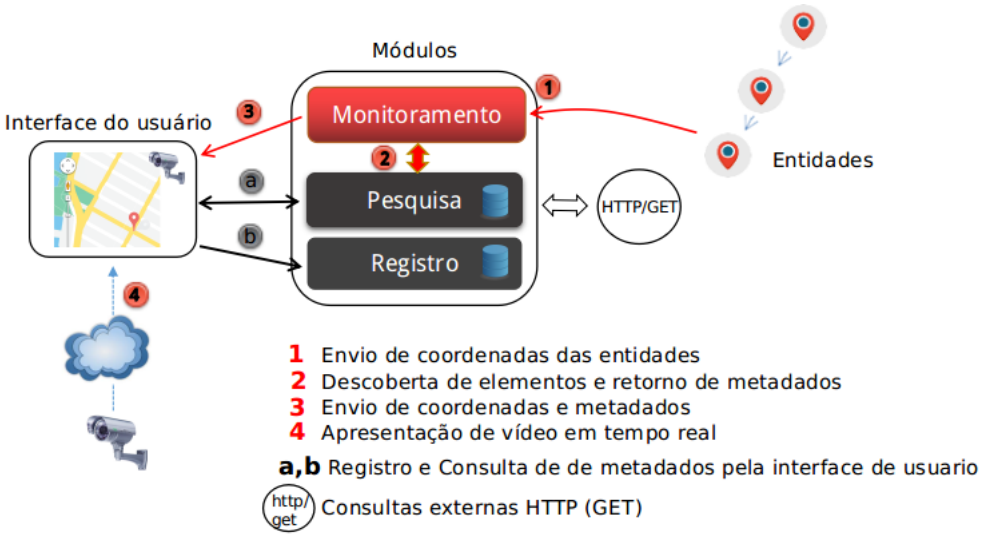


Figura 4.2: Etapas dos módulos

4.2.1 Módulo de Registro

Na Figura 4.3 é mostrado o módulo de registro. Este módulo é responsável por registrar as câmeras e os seus metadados, no qual estes metadados são preenchidos através de uma interface de usuário e gravados em um banco de dados.

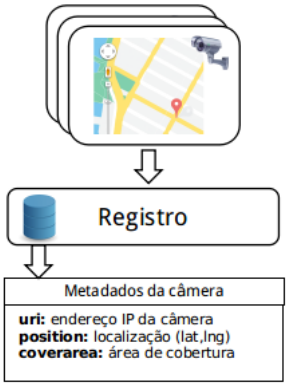


Figura 4.3: Registro de metadados

O banco de dados é um local de armazenagem de metadados relacionados a elementos físicos ou não, mas que possam ser encontrados na Internet. O modelo de metatados da

câmera pode ser adaptado para outros tipos de sensores, contudo, para este trabalho, serão utilizadas apenas câmeras IP.

4.2.2 Módulo de Consulta

O módulo de consulta mostrado na Figura 4.4 mantém a comunicação com o banco de dados de metadados de câmeras e sua tarefa é verificar se existe área de cobertura no local geográfico enviado pelo módulo de monitoramento, pela interface de usuário através de um mapa e externamente por uma chamada HTTP/GET. Caso metadados sejam encontrados serão encaminhados para o que fez a chamada de consulta. A chamada externa habilitada no módulo de consulta permite que consultas sejam realizadas por qualquer navegador ou comando de sistemas externos utilizando o protocolo HTTP para descobrir câmeras dada uma região.

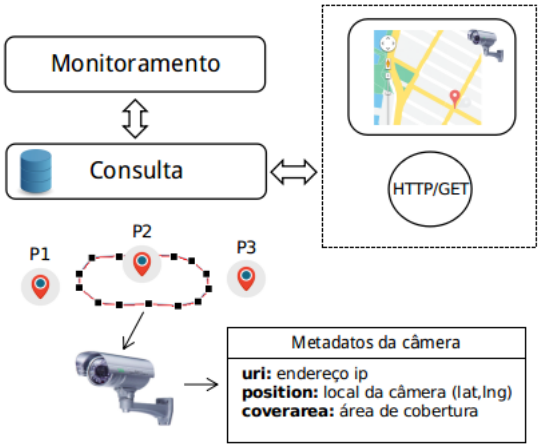


Figura 4.4: Consulta de metadados

4.2.3 Módulo de Monitoramento

Conforme mostrado na Figura 4.5 as coordenadas indicadas (P1,P2,P3) são enviadas ao módulo de monitoramento para consulta no bando de dados de metadados de câmeras. As interfaces de usuários (podem ser várias) são conectadas ao módulo de monitoramento para receberem todas as alterações de posição das entidades e consequentemente a obtenção de novas imagens das câmeras pela transposição de várias áreas de cobertura.

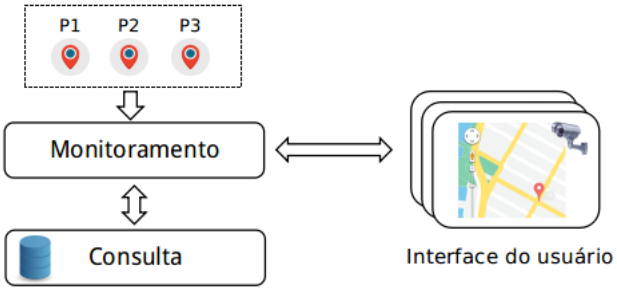


Figura 4.5: Consulta de metadados

4.3 Informações sobre a câmera e área de cobertura

Uma forma direta para compor ou desenhar as áreas de cobertura seria pelo método totalmente gráfico na própria tela do computador enquanto se visualiza as imagens da câmera. Para produzir os contornos destas áreas de cobertura em um monitor, como mostrado na Figura 4.6, é preciso realizar uma transformação de coordenadas do universo de pixels da tela para as coordenadas físicas do local. Um contorno virtual formado pelo conjunto destes pixels corresponderá a uma área na superfície da Terra por onde o usuário ou entidade transitará ativando a câmera. Para isso é necessário converter as coordenadas de pixel (u,v) para as coordenadas geográficas (latitude, longitude). Um trabalho de mapeamento de coordenadas foi realizado em [34] com o registro de cinco posições.

Com as informações de localização espacial da câmera, altura e ângulo de inclinação, é possível realizar cálculos com boa aproximação para descobrir as coordenadas do centro da área de cobertura na Figura 4.6. Com obtenção das coordenadas do centro e um raio de alcance têm-se os parâmetros de uma área de cobertura, contudo, do ponto de vista da câmera, o campo de visada (FoV - *Field of View*) pode estar obstruído e reduzido, tornando o trabalho de mapeamento mais complexo do que simples formas circulares.

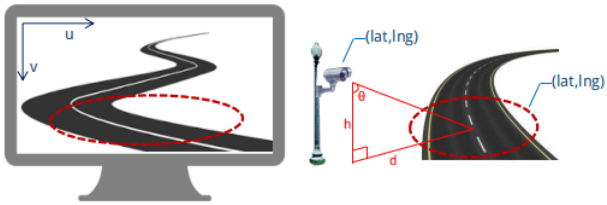


Figura 4.6: Definição de área de cobertura

Em [53] é proposto um método em tempo real para determinar a distância de um objeto até as câmeras através da modificação física do equipamento com a instalação de um sensor no eixo óptico da lente. Propõe também a utilização de uma função de

definição para determinar quando a imagem se torna mais nítida de acordo com o ângulo de inclinação do sensor. Este método poderia determinar a distância de uma região ou objeto de referência à câmera, e a partir desta distância calcular a localização geográfica do objeto e do seu entorno com base na localização da própria câmera. Contudo, por questões técnicas, as alterações físicas dos equipamentos de modelos diferentes tornaria o método inviável.

Um modelo de representação de áreas de cobertura pode ser definido através de áreas de figuras geométricas construídas por vértices interligados no qual cada vértice traz em si informações associadas a coordenadas geográficas. O sistema geodésico WGS84 (*World Geodetic System*) foi proposto em 1984 e é composto de um sistema de coordenadas para a Terra com uma superfície elipsoide de referência. Com esse sistema, pode-se determinar a localização de um ponto na superfície da Terra sem a necessidade de fazer qualquer tipo de ajuste ou calibragem em câmeras.

A Figura 4.7a ilustra em um exemplo de geometria representando a área de cobertura de uma câmera na Figura 4.7b. Conseguindo associar cada vértice do polígono a um par de coordenadas latitude e longitude, o conjunto destes vértices pode representar fisicamente a localização de uma área. Como mostrado na Figura 4.7, o posicionamento dos vértices em um mapa é baseado na própria percepção do usuário da extensão visual da câmera.

Outras informações devem estar associadas à câmera além da própria área de cobertura. Estas informações deverão ser metadados importantes para a identificação da câmera e todo o contexto que ela representa. Definem-se como metadados um identificador único da câmera, sua localização geográfica, seu endereço físico, endereço de localização na Internet ou URI, área de cobertura definida pelo conjunto de localizações reais dos vértices da figura geométrica escolhida, um nome genérico para a câmera, nome do proprietário ou de quem for o responsável e uma opção de escolha para definir se a câmera terá acesso público ou particular.

(a) Polígono sobre mapa (b) Imagem da câmera

Figura 4.7: Determinação da área de cobertura através de um polígono.

4.4 Servidor de vídeo

O stream de vídeo da câmera ao cliente é do tipo transmissão ao vivo, ou seja, todo o conteúdo gerado é imediatamente encaminhado ao cliente para que o monitoramento em tempo real ocorra o mais rápido possível. Através da apresentação do vídeo na plataforma web é facilitado o alcance de utilização em um número maior de dispositivos, como tablets ou smartphones.

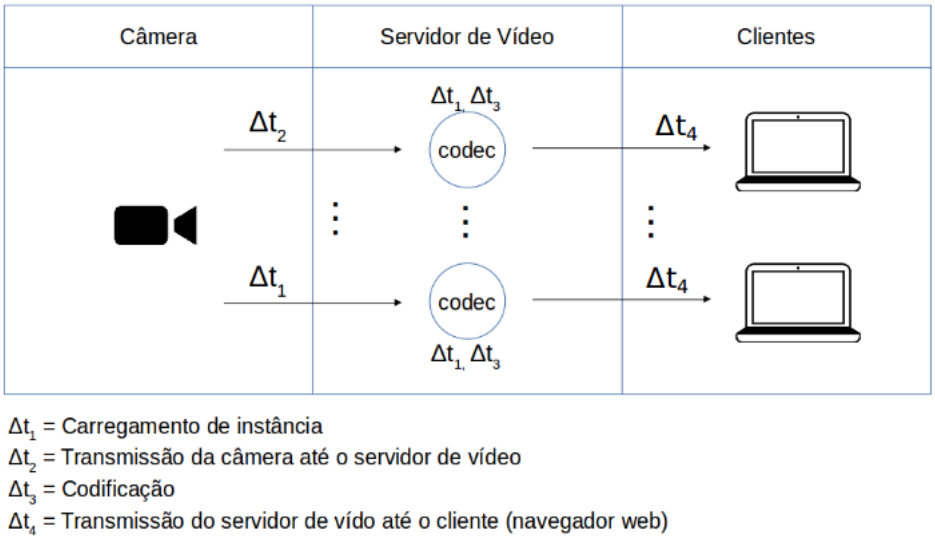
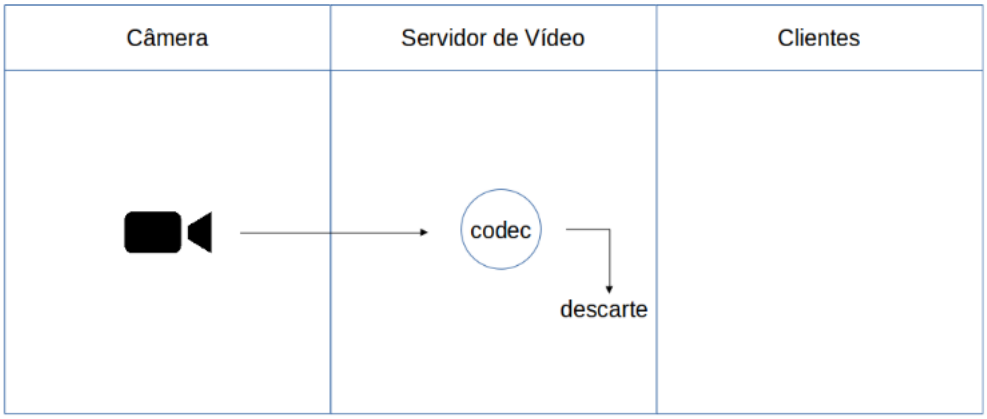


Figura 4.8: Várias instâncias interligando logicamente o cliente à camera

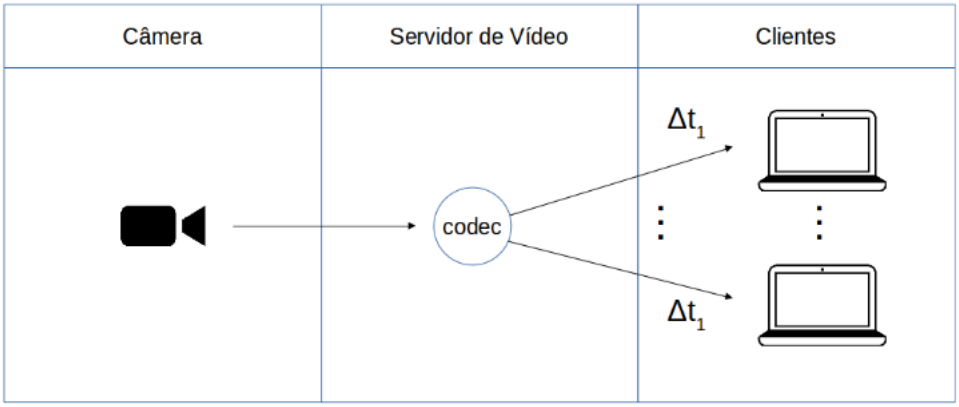
Na Figura 4.8, temos o servidor de vídeo gerando pacotes codificados para cada requisição do cliente. Neste processo, para cada requisição do cliente, uma nova instância é gerada. Esta instância troca informação com a câmera para receber um fluxo de bits, que é transcodificado para o formato desejado e encaminhado para o cliente específico. A instância é representada pelo círculo. O Δt_1 significa o tempo que leva para a instância ser alocada na memória. Δt_2 é o tempo que leva para as trocas de informações entre o servidor de vídeo e a câmera, bem como o recebimento do fluxo de bits das imagens e também o tempo gasto na transcodificação. Δt_3 é o tempo gasto para a codificação, e Δt_4 é o tempo gasto para enviar os bits de vídeo para o cliente. Neste caso a relação é de 1 para 1, ou seja, para cada instância, existe um cliente requisitando imagens. A desvantagem deste método é a dificuldade de escalar o sistema, pois quando muitos usuários estão requisitando imagens é necessário carregar uma instância para cada um.

Na Figura 4.9a, não existe cliente, por isso o fluxo de bits que o servidor de vídeo deveria encaminhar é descartado até que alguma requisição de vídeo aconteça. O cenário muda em (b) quando novos clientes surgem. Nesta etapa, os bits que eram descartados

passam a ser transferidos para todos os clientes, ou seja, existe apenas um fluxo entre o servidor e a câmera com a relação de 1 para n. A vantagem é a de conseguir obter maior escalabilidade por utilizar apenas uma instância por câmera, e também em ganhar tempo, pois os bits já estão prontos para a entrega, por isso este método será utilizado neste trabalho. A desvantagem para o método anterior é a necessidade de se manter o uso contínuo da rede.



(a) Nenhum cliente requisitando imagens



Δt_1 = Transmissão do servidor de vído até o cliente (navegador web)

(b) Vários clientes requisitando imagens.

Figura 4.9: Modelo onde é necessário apenas uma instância para n clientes.

4.5 Interface do usuário

A interface com o usuário é uma página web conectada ao Servidor de monitoramento e tem basicamente duas finalidades. A primeira é realizar o monitoramento geográfico e visual das entidades através do recebimento de metadados do módulo de consulta. A segunda finalidade pode ser dividida em desenho e cadastro de áreas de cobertura, e de informações complementares como o endereço IP da câmera, etc. Os dados de cadastro são enviados para o módulo de registro, que por sua vez faz uma interface com o banco de dados.

Capítulo 5

Protótipo

Neste capítulo é descrita a implentação do protótipo utilizado para a avalição da arquitetura proposta apresentada no Capítulo 4, como parte dos conceitos multi-metodológicos da Seção 1.3 para a experimentação necessária no desenvolvimento de sistemas.

5.1 Servidor de monitoramento

O servidor de monitoramento é composto pelos módulos descritos na Seção 4.2. Nesta seção são descritos detalhadamente os recursos utilizados para a implementação de cada um deles. No módulo de registro da Seção 5.1.1, é apresentado um esquema de campos e valores de metadados e a verificação das possibilidades de gravação das informações em banco de dados. No Módulo de consulta da Seção 5.1.2, são abordados procedimentos para a consulta de metadados e, na Seção 5.1.3 sobre o Módulo de monitoramento, é apresentado o método utilizado para a realização de envio de mensagens de forma rápida contendo as localizações das entidades.

5.1.1 Módulo de registro

O esquema de campos e valores para armazenamento das câmeras é mostrado na Tabela 5.1. Neste formato de documento, são mantidas as informações do nome do usuário (*user*), nome da câmera (*camname*), local da câmera (*position*), sua área de cobertura (*coverarea*), URI de acesso para localizar a câmera na Internet, um campo de publicidade (*public*) contendo um valor *True* ou *False* para permitir ou não que a câmera seja vista por outros usuários e o campo de endereço da câmera (*address*). O campo id é um identificador único.

Tabela 5.1: Formato do documento

Campo	**Descrição**
id:	Identificador único
user:	Quem fez o cadastro
camname:	Nome da câmera
position:	Local da câmera
coverarea:	Area de cobertura
uri:	Link de acesso
public:	Visível ou não para outros
address:	Endereço real da câmera

A escolha por um banco de dados para este trabalho foi baseada principalmente na capacidade de armazenar estruturas de dados geoespaciais (GeoJSON)[1] para a importação e exportação de áreas poligonais para os mapas. O formato GeoJSON define objetos JSON (*JavaScript Object Notation*) e a maneira na qual eles são combinados para representar características geográficas usando o sistema WGS84. Outras características essenciais no banco de dados além de armazenar coordenadas geográficas são calcular distância e verificar proximidades de áreas poligonais para atender a exigência de se fazer buscas por áreas de coberturas. Um banco de dados que consiga trabalhar bem com um grande volume de dados e, ao mesmo tempo, de fazer buscas em curtos intervalos de tempo também são requisitos indispensáveis.

Bancos de dados orientados a documentos são úteis em aplicações modernas com alta escalabilidade de dados, em que esquemas são alterados para a complementação de novos recursos ou alteração de tipos de dados. De acordo com Jing Han [23] um banco de dados relacional por sua complexidade lógica é propenso a acarretar declínios de eficiência de leitura e escrita quando há um aumento da base de dados, portanto, após uma breve revisão de tipos de banco de dados NoSQL (*Not Only SQL*) orientado a documentos, com suporte a GeoJSON, de código aberto, com boa documentação e popularidade, foi escolhido o MongoDB [37].

MongoDB é um banco de dados orientado a documentos, que contorna as dificuldades dos bancos de dados existentes como escalabilidade e agilidade. Também conhecido popularmente por ser NoSQL e por seguir um modelo diferente dos bancos de dados tradicionais relacionais que lidam com registros, onde tudo é representado de forma bidimensional com linhas e colunas. Em um modelo de banco de dados relacional é necessário

[1] http://geojson.org/

que seja definido previamente um esquema antes de adicionar os dados, ou seja, o banco de dados precisa saber antecipadamente o que será armazenado e isso dificultaria em atualizações nos modelos de metadados para diferentes tipos de elementos. De acordo com [10], MongoDB não tenta ser tudo para todos, mas tenta alcançar um equilíbrio entre recursos e complexidade. Possui os recursos que realmente importam para a grande maioria das aplicações Web atuais como índices, replicação, fragmentação, uma sintaxe de consulta avançada e um modelo de dados muito flexível e sem sacrificar a velocidade. Os dados armazenados por este banco de dados são no formato *Binary JSON* (BSON)[2] que é um formato binário, em que pares ordenados de chaves e valores são armazenados como uma entidade única que é similar a um objeto JSON como mostrado na Figura 5.1. É permitido incorporar aos valores dos campos outros documentos, *arrays* e *arrays* de outros documentos.

```
{
  name: "sue",                    ← field: value
  age: 26,                        ← field: value
  status: "A",                    ← field: value
  groups: [ "news", "sports" ]    ← field: value
}
```

Figura 5.1: Formato Binary JSON (BSON)

Com o suporte para GeoJSON, é possível realizar operações de pesquisa sobre os tipos de objetos como: *Point*, *LineString*, *Polygon*, *MultiPoint*, *MultiLineString*, *MultiPolygon*, *GeometryCollection*. Na Figura 5.2, seguem dois exemplos de documentos GeoJSON. O campo *coordinates* se refere às coordenadas que pertencem ao tipo de objeto utilizado.

```
{
type: "Point",
coordinates: [-73.856077, 40.848447]
}
```

(a) Tipo ponto (Point)

```
{
type: "Polygon",
coordinates: [[0,0],[3,6],[6,1],[0,0]]
}
```

(b) Tipo polígono (Polygon)

Figura 5.2: Exemplo de documentos GeoJSON para o tipo ponto e polígono

5.1.2 Módulo de consulta

São possíveis operações de pesquisa ou consulta no MongoDB com dados geoespaciais em objetos GeoJSON através de operadores, como *$near*, *$geoIntersects* e *$geoWithin*. Coordenadas Euclidianas desde que indexadas corretamente também são possíveis de

[2]https://docs.mongodb.com/manual

armazenar e de se efetuar cálculos e são chamadas de pares de coordenadas legadas. A Tabela 5.2 mostra um resumo de alguns operadores e o que é retornado na pesquisa.

Tabela 5.2: Exemplos de operadores de pesquisa

Operador	**Retorno de documentos**
$near	próximos a um ponto
$geoIntersects	em interseção com outros objetos
$geoWithin	internos a um contorno

Quando uma pesquisa é realizada usando coordenadas de latitude e longitude, o banco de dados verifica quais documentos possuem estes parâmetros dentro do campo que guarda as regiões das áreas geográficas. Um exemplo se encontra na Figura 5.3, em que uma coordenada de latitude e longitude está dentro da região de um polígono.

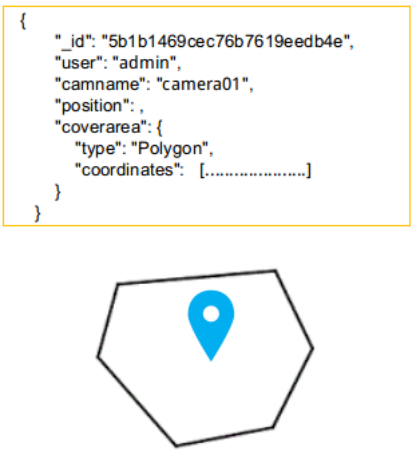


Figura 5.3: Representação de documento e interceptado por uma coordenada geográfica.

As operações de pesquisa ocorrem sempre que uma entidade envia suas coordenadas para o Módulo de comunicação da Figura 4.1. Também é realizada uma consulta de elementos quando uma requisição HTTP com o método GET é recebida pelo módulo contendo na URL as coordenadas e um raio de cobertura.

5.1.3 Módulo de monitoramento

Para o bom funcionamento do monitoramento, é necessário o envio rápido de coordenadas de GPS da entidade para o módulo de comunicação. O protocolo Websocket [16] foi usado nos terminais das entidades que portavam um smartphone com conexão a Internet para envio das localizações para o servidor. O servidor Websocket foi implementado sobre o NodeJS [49] que é um interpretador de *JavaScript*. A opção de utilização do

NodeJS também se deu em favor de uma padronização de linguagem de programação tanto pelas páginas Web no lado do cliente (*Front-End*) e no lado servidor (*Back-End*), incluindo a facilidade do NodeJS de se conectar com o banco de dados MongoDB. Além disso, o comparativo realizado entre NodeJS, PHP e Python-web para aplicações de rede com uso intenso demonstrou um ótimo desempenho para o NodeJS [32]. Na Figura 5.4, as entidades monitoradas enviam suas coordenadas de localização por mensagens e em seguida encaminhadas para consulta no banco de dados de elementos. O Websocket *Server* mantém uma conexão com a Interface do usuário (Página Web), que recebe mensagens contendo as localizações e as áreas de cobertura encontradas, por exemplo, no caso do *User2* haverá a identificação de câmera e área de cobertura.

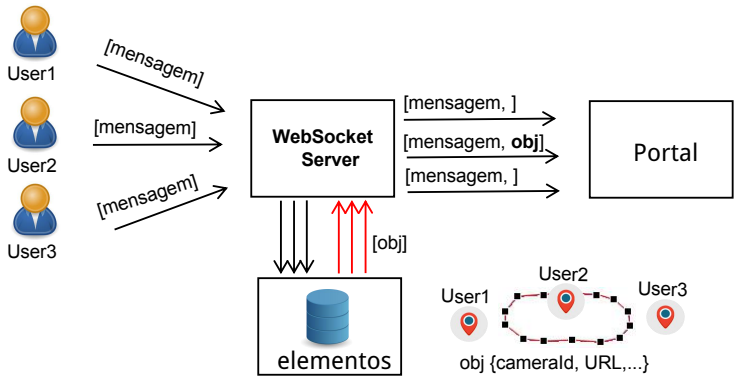


Figura 5.4: Servidor WebSocket recebendo mensagens e realizando pesquisas

5.2 Servidor de vídeo

Conforme os conceitos introduzidos sobre *streaming* de vídeo na Seção 2.3, existem basicamente três classes de aplicações que manipulam vídeo: fluxo de vídeo contínuo armazenado, voz e vídeo interativos e fluxo de vídeo contínuo ao vivo. Para a transmissão de vídeo de uma câmera, é necessário que a transmissão seja do tipo fluxo contínuo ao vivo, ou seja, todo o conteúdo gerado é imediatamente encaminhado ao cliente para que o monitoramento em tempo real aconteça da forma mais rápida possível. Na Seção 2.3, também são relatados alguns protocolos para transmissão de vídeo na Internet, especificamente sobre o protocolo HTTP como HLS e MPEG-DASH. Contudo, existe um tempo de processamento para a montagem dos arquivos de mídia e índices oriundos destas técnicas. Este tempo de geração de arquivos acaba dificultando sua aplicação para um monitoramento onde se deseja acompanhar uma entidade com o menor atraso possível. Verificou-se que para a câmera usada nos testes, os protocolos e formatos habilitados são os seguintes:

- rtsp://ip:port/video.h264
- rtsp://ip:port/video.mp4
- rtsp://ip:port/video.mjpg
- http://ip:port/video.mjpg
- rtsp://ip:port/video.3gp

Como pode ser visto, não existe uma opção do tipo *http://ip:port/video.mp4*, sendo o protocolo RTSP o padrão mais utilizado na listagem acima com a exceção do HTTP, que pode transmitir na codificação MJPG (*Motion JPEG*). Contudo, devido à incompatibilidade de uso direto do RTSP em uma página Web, é necessária uma técnica diferente, como por exemplo, a utilização de algum servidor de mídia que receba o fluxo de vídeo da câmera em RTSP e transforme em algo compatível para o navegador. Existem diversos softwares para serviços de *streaming* de vídeo disponíveis, mas poucos são de uso livre e que funcionem a contento. A Tabela 5.3 apresenta uma pequena lista de *softwares* para Linux. Existem outros *softwares* de *streaming* de vídeo, porém, os que não eram *Open Source* não foram testados.

No desenvolvimento desta pesquisa não foi encontrada uma ferramenta que atendesse o requisito de baixa latência para a transmissão de vídeo, logo, optou-se por criar um servidor de vídeo próprio baseando-se no FFMPEG, que é um *framework* multimídia capaz de codificar, transcodificar, decodificar, multiplexar, filtrar e reproduzir uma enorme variedade de formatos de áudio e vídeo. Considerando as formas apresentadas na Figura 4.8 e Figura 4.9 foram elaborados dois métodos para transmissão de vídeo.

Os formatos de áudio e vídeo mais comuns para os navegadores Web são informados na Tabela 5.4. Para MP4, os codecs de vídeo e áudio são H264 e AAC, para webM, os codecs são VP8 e Vorbis e, para Ogg, os codecs Theora e Vorbis[6].

Como pode ser visto na Tabela 5.4, o formato de container MP4 é reconhecido pelos navegadores mais populares e em seguida,porém menos compatíveis os formatos webM e Ogg. O formato mais apropriado, de acordo com essa informações, deveria ser o MP4. Contudo, os testes com MP4 fragmentado não se saíram muito bem com a técnica da Figura 4.9. A opção de transferência em MJPEG está disponível na câmera e pode ser utilizada através da TAG no código HTML. Isto evitaria a criação de uma ferramenta extra para o serviço de transcodificação e envio de bits para o navegador (cliente). Contudo,

[6]www.w3schools.com

Tabela 5.3: Exemplos de softwares para *streaming* de vídeo

Sistema de Streaming	**Comentários**
FFserver[3]	Trabalha em conjunto com FFmpeg e utiliza vários formatos como mpeg, mpegvideo, ogg, mpjpeg, asf, etc. Mostrou-se ineficiente para a maioria dos formatos. A aplicação é finalizada em momentos aleatórios com pouca geração de informação. Este software foi descontinuado em janeiro de 2018.
LIVE555 Media Server[4]	É um servidor de mídia que pode ler arquivos de vários formatos e transmiti-los por RTP ou RTSP. Não será útil pois o que se deseja é o inverso, um servidor que leia de fontes RTSP e converta para algo compatível com páginas Web.
NGINX[5]	Na sua versão Open Source, o nginx (Engine x) é um servidor HTTP (entre outras coisas como proxy reverso, etc). Consegue fazer *streaming* para o formato FLV através do módulo nginx-rtmp-module. O módulo recebe um fluxo RTMP (*Real-Time Messaging Protocol*) que foi desenvolvido para a transmissão de áudio, vídeo e dados para as plataformas da Adobe Flash. Este fluxo é transformado para HLS e encaminhado para o cliente Web. Como comentado ainda nesta seção, este método não é o melhor para acompanhamento em tempo real pelo atraso na geração dos arquivos. Sem contar que a câmera trabalha essencialmente com o protocolo RTSP, logo, foi necessário uma conversão prévia para RTMP com o FFmpeg aumentando mais ainda o tempo de envio para o cliente.

Tabela 5.4: Formatos suportados pela tag <video>

Navegador	**MP4**	**webM**	**Ogg**
Internet Explorer	sim	não	não
Chrome	sim	sim	sim
Firefox	sim	sim	sim
Safari	sim	não	não
Opera	sim	sim	sim

é necessário realizar uma aferição do consumo de banda, pois isso pode prejudicar a escalabilidade do sistema proposto. Esse teste é descrito no Capítulo 5.

Através de uma pesquisa sobre o formato Ogg[7], viu-se a possibilidade de aceitação do navegador (cliente) deste fluxo de vídeo, desde que utilizando os primeiros bits do fluxo,

[7] www.xiph.org/ogg/doc/rfc3533.txt

pois estes bits iniciais contêm as informações para que o navegador identifique corretamente o que será consumindo. O formato de encapsulamento Ogg é primordialmente um formato para *streaming* de mídia, para uso livre, e capaz de encapsular qualquer quantidade de áudio e video e áudio codificado, como também outros streams de dados em um único stream de bits. A codificação provê um fluxo de bits que é subdividido em pacotes, e o processo de encapsulamento divide os pacotes em segmentos que se agrupam em uma ou mais páginas de tamanhos variados. O fluxo de bits no formato Ogg é chamado de fluxo físico, e consiste na multiplexação de diversas páginas incluindo as páginas advindas de outros fluxos lógicos de bits resultando em um conjunto de páginas concatenadas.

O cabeçalho de uma página contêm todas as informações necessárias para fazer a demultiplexação e reconstruir o fluxo lógico de bits executando também a recuperação de erros, entre outras tarefas. O mais importante é conseguir capturar estas informações de cabeçalho e enviar para a página Web do cliente para reconhecer o formato do fluxo de vídeo. Abaixo segue um exemplo de cabeçalho na Figura 5.5.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1| Byte
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| capture_pattern: Magic number for page start "OggS"           | 0-3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| version       | header_type   | granule_position              | 4-7
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               | 8-11
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               | bitstream_serial_number       | 12-15
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               | page_sequence_number          | 16-19
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               | CRC_checksum                  | 20-23
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               |page_segments  | segment_table | 24-27
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ...                                                           | 28-
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figura 5.5: Cabeçalho teórico Ogg

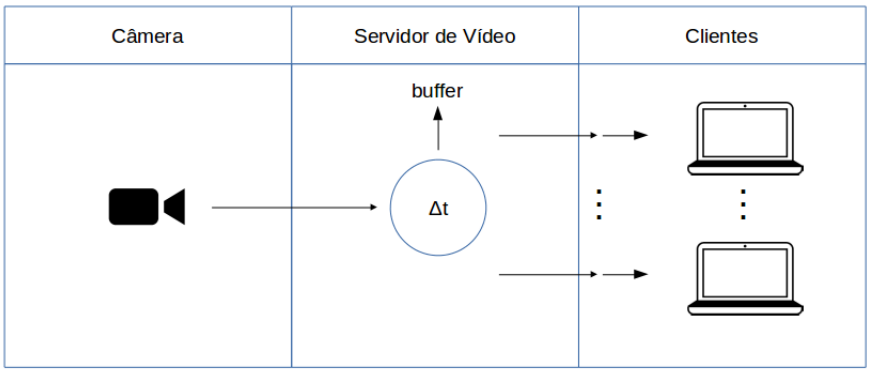
Na Figura 5.6, tem-se um exemplo de cabeçalho real que foi armazenado em um *buffer* com os primeiros bits do fluxo de vídeo. Como pode-se observar, existe um padrão em Hexadecimal 0x4f 0x67 0x67 0x53 identificando o container Ogg, e mais adiante temos a identificação de cabeçalho Theora logo após 0x80. Abaixo segue um exemplo da conversão do fluxo da câmera para o formato Ogg e codec Theora.

"ffmpeg -stimeout 5000000 -rtsp_transport tcp -nostdin -loglevel error -i rtsp://ip:port/video.mp4 -c:v libtheora -an -q:v 7 -f ogg pipe:1"

Os bits gerados pelo FFMPEG são direcionados para o cliente através da última parte do comando (pipe:1), onde o 1 significa que está utilizando a saída padrão do processo em

```
4F 67 67 53 00 02 00 00 00 00 00 00 00 00 08 00 3E 37 OggS............>7
00 00 00 00 62 21 AC 67 01 2A 80 74 68 65 6F 72 61 03 ....b!.g.*.theora.
02 01 00 28 00 1E 00 02 80 00 01 E0 00 00 00 00 00 19 ...(..............
00 00 00 01 00 00 01 00 00 01 00 00 00 00 B0 C0 4F 67 ................Og
67 53 00 00 00 00 00 00 00 00 00 00 08 00 3E 37 01 00 gS............>7..
00 00 E6 30 B7 69 0E 71 FF FF FF FF FF FF FF FF FF FF ...0.i.q..........
FF FF 90 81 74 68 65 6F 72 61 0D 00 00 00 4C 61 76 66 ....theora....Lavf
35 37 2E 38 33 2E 31 30 30 03 00 00 00 20 00 00 00 65 57.83.100.... ...e
6E 63 6F 64 65 72 3D 4C 61 76 63 35 37 2E 31 30 37 2E ncoder=Lavc57.107.
31 30 30 20 6C 69 62 74 68 65 6F 72 61 0D 00 00 00 74 100 libtheora....t
69 74 6C 65 3D 51 53 74 72 65 61 6D 1C 00 00 00 44 45 itle=QStream....DE
53 43 52 49 50 54 49 4F 4E 3D 51 53 74 72 65 61 6D 69 SCRIPTION=QStreami
6E 67 20 4D 65 64 69 61 82 74 68 65 6F 72 61 BE CD 28 ng Media.theora..(
F7 B9 CD 6B 18 B5 A9 49 4A 10 73 9C E6 31 8C 52 94 A4 ...k...IJ.s..1.R..
21 08 31 8C 62 10 84 21 08 40 00 00 00 00 00 00 00 00 !.1.b..!.@........
00 11 6D AE 53 67 92 C8 FC 56 12 FC 78 39 5B 6C E6 2A ..m.Sg...V..x9[l.*
```

Figura 5.6: Cabeçalho de um fluxo real



Δt = carregamento de instância + codificação

Figura 5.7: Esquema atualizado para servidor de vídeo

execução. O cliente após fazer a requisição de vídeo recebe primeiro os bits do buffer e em seguida o vídeo transcodificado. Na Figura 5.7, é mostrada a atualização esquemática utilizando o buffer.

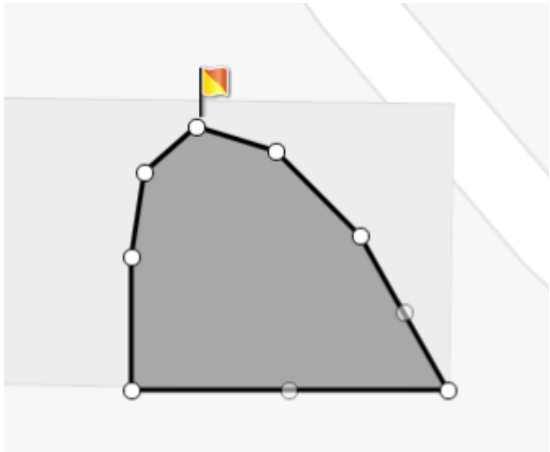
5.3 Interface do usuário

5.3.1 Desenho da área de cobertura

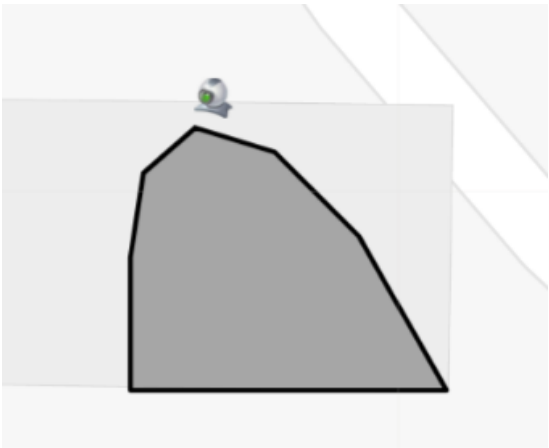
Uma opção gráfica para determinar pontos de latitude e longitude sobre o vértice de forma simplificada seria diretamente sobre mapa com suporte ao sistema geodésico WGS84 (*World Geodetic System*). A empresa Google oferece um serviço gratuito permitindo a desenvolvedores incorporar mapas nas páginas Web utilizando o padrão WGS84 em conjunto com o tipo de projeção *Mercator* para que um ponto no mapa possa representar uma coordenada mundial. Outros serviços conhecidos de mapas também estão disponí-

veis, como Bings Maps e OpenStreetMap. Após a pesquisa apresentada na Seção 2.2.3, optou-se para a escolha da plataforma Google Maps. O Google Maps permite que seja adicionado sobre o mapa uma classe de objeto chamada Dados. Nesta classe, pode-se utilizar marcadores, linhas, polígonos e ainda exportar estes objetos no formato GeoJSON [8] usando notação JSON (*JavaScript Object Notation*). Contudo, não tem nessa classe uma ferramenta própria de desenho de polígonos, e para tal existe uma outra classe chamada de *DrawingManager* que pertence à biblioteca *drawing*. A Figura 5.8 mostra a edição de uma câmera e sua área de cobertura usando a ferramenta de desenho.

(a) Drawing Control



(b) Edição de polígono e marcador



(c) Polígono e marcador na camada de Dados

Figura 5.8: Exemplo de edição. a) Ferramenta de desenho Drawing Control. b) Inserção de marcador e vértices do polígono. c) Marcador e polígono na camada de Dados

Após os desenhos serem concluídos com o *DrawingManager*, foi necessário então, copiar todos os itens construídos para camada de Dados e, em seguida, exportá-los para o formato GeoJSON. O *DrawingManager* apesar de útil por já ser uma ferramenta pronta não pode ter seus componentes alterados, ou seja, não é possível colocar um botão extra para outras funções, por exemplo, de salvar a posição da câmera e da área de cobertura no bando de dados. Para resolver esta questão nesta dissertação, foi desenvolvida uma ferramenta própria de desenho de polígonos e de inserção de câmeras mostrada na Figura 5.9. Após as etapas de posicionamento da câmera no mapa e a definição de área de cobertura, já é possível enviar tudo automaticamente para o banco de dados. A Figura 5.10 mostra a busca de um endereço em modo texto para a adição de uma câmera no local, e na mesma imagem aparece outra câmera já cadastrada. Na Figura 5.11, é adicionado o marcador e desenhada a área de cobertura. Na Figura 5.12, são adicionados o nome

[8] http://geojson.org/

da câmera e o IP (ou URL) de acesso. Os outros campos são informações entregues pela ferramenta de *Geocoding* baseada na latitude e longitude da câmera. A Figura 5.13 mostra a mudança de cor da área indicando que os dados foram registrados com sucesso.

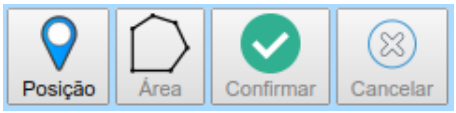


Figura 5.9: Novo controle para adicionar câmera e área de cobertura.

Figura 5.10: Procura por endereço no mapa

Figura 5.11: Desenho da área de cobertura

Figura 5.12: Formulário para preenchimento de metadados da câmera

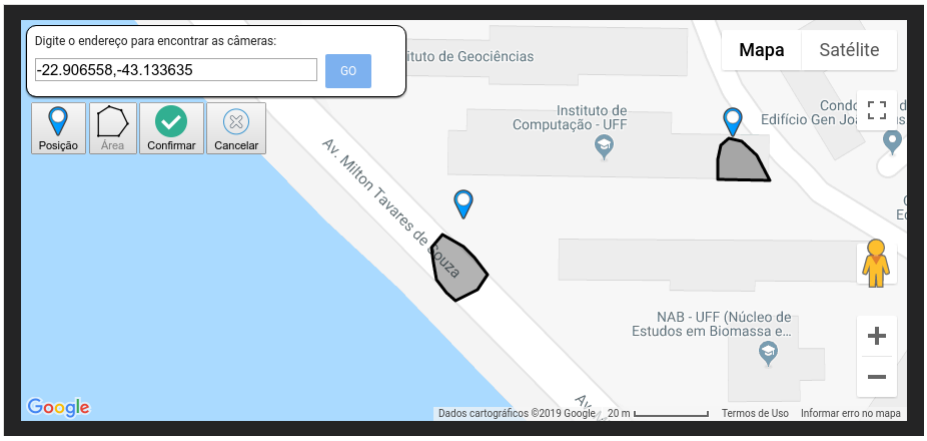


Figura 5.13: Metadados registrados

5.3.2 Cadastro da área de cobertura

Os elementos adicionados sobre o mapa (Marcador e Polígono) representam a câmera e sua área de cobertura. Eles são adicionados pela ferramenta criada para edição, mas precisam ser organizados para se adequarem ao formato de documento do banco de dados. Estes elementos são independentes, ou seja, não existe associação entre eles, logo foi construída uma ligação entre Marcador e Polígono para informar que eles pertencem a mesma câmera. Esta ligação pode ser realizada com o campo de propriedades mostrado na Figura 5.14(a) e 5.14(b). O campo de propriedades permite adicionar informações extras aos objetos, deste modo, no momento da criação do elemento as informações de controle são adicionadas em suas propriedades como motrado em pontilhado. A Figura 5.14(c) mostra a junção destes dois elementos para formar uma estrutura compatível com o formato indicado na Tabela 5.1 que em seguida é enviada para a função de cadastro do Módulo de comunicação.

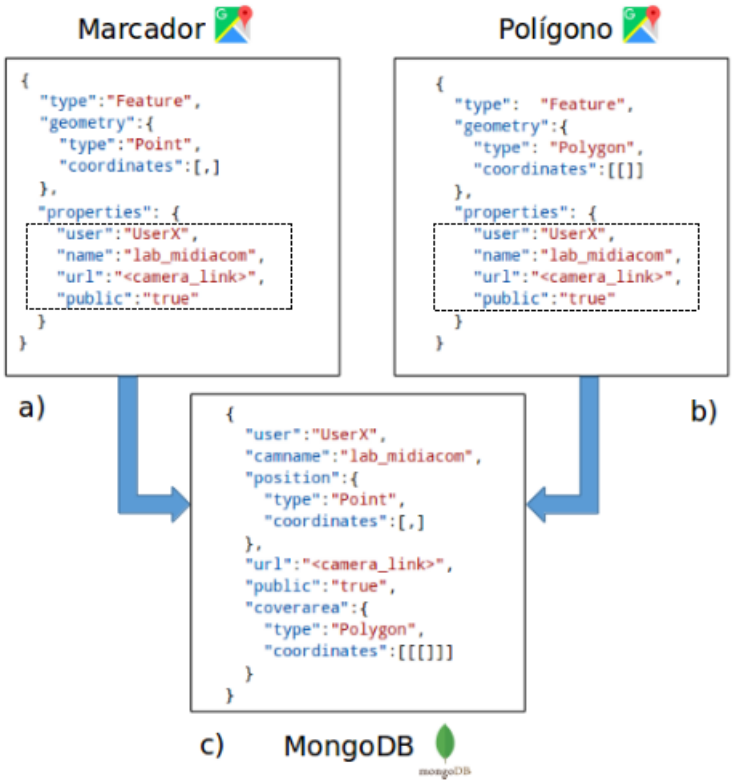


Figura 5.14: a) Elemento do tipo Point representando a posição da câmera no mapa. (b) Polígono representando a área de cobertura onde cada vértice possui uma coordenada de latitude e longitude. c) Junção dos dois objetos.

Capítulo 6

Avaliação Experimental

Neste capítulo, como parte da prova de conceito, á apresentada a avaliação do protótipo da arquitetura proposta no Capítulo 4. Como parte do cenário de teste é realizado o monitoramento de um usuário portando um celular com GPS. Também foi realizada uma consulta de elementos com uma operação HTTP (método GET) através de uma URL. Medições de taxas de transferência de vídeo são feitas para algumas categorias de qualidade da câmera, e métodos de transmissão de vídeo são abordados e testados para calcular o atraso de envio da imagem desde a captura da câmera até sua exibição no cliente.

6.1 Cenário

No cenário da Figura 6.1, as máquinas servidoras estão em laboratórios distintos dentro da UFF. Uma máquina cliente está situada na mesma cidade, mas fora da rede do campus, acessando a interface do usuário por meio de Internet residencial e modem Wifi. Uma das máquinas, chamada Servidor de monitoramento e Web, possui o banco de dados de metadados, Módulos de registro, consulta, monitoramento e a interface de usuário. O servidor de vídeo é uma máquina virtual localizada em outro laboratório e faz somente o processamento de vídeo. Na máquina virtual, foi instalado o Nginx[1] para transmitir vídeo usando o protocolo de comunicação para streaming de mídia HLS (HTTP *Live Streaming*) com codificação para o formato Flash Video (FLV). Na mesma máquina virtual, foi usado o servidor de vídeo proposto na Seção 5.2.

O monitoramento de entidades é apresentado na Figura 6.2b, em que uma janela é

[1]https://www.nginx.com

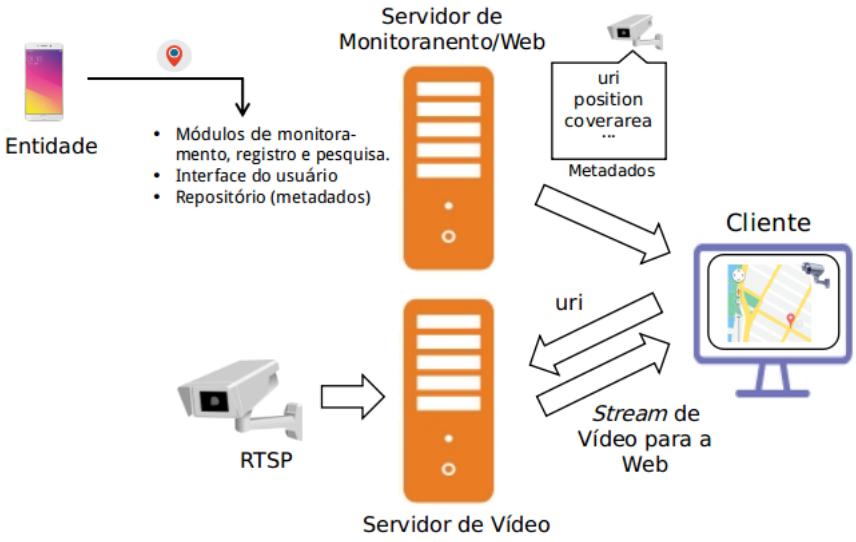
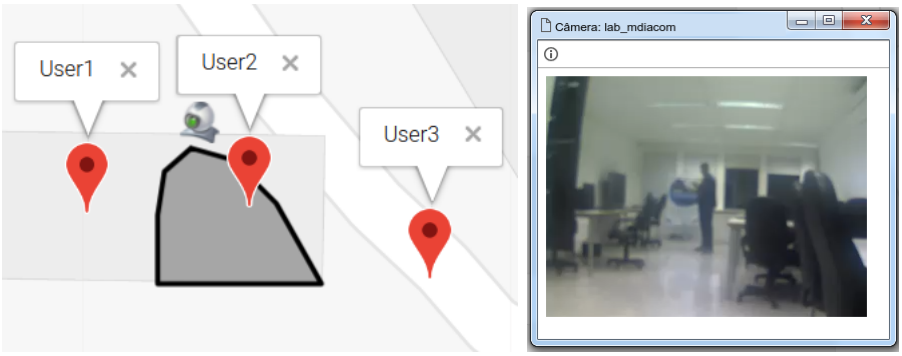


Figura 6.1: Cenário de monitoramento

acionada no momento em que o usuário chamado *User2* ultrapassa os limites do polígono que representa a área de cobertura da câmera da Figura 6.2a. Neste exemplo, a câmera se encontra *indoor* para ser possível ver melhor o usuário, pois no teste *outdoor* o zoom da câmera foi insuficiente.



(a) (b)

Figura 6.2: Envio de informações de monitoramento para a interface de usuário (cliente)

Na Figura 6.3, é mostrado um exemplo de retorno de uma pesquisa realizada pelo módulo de monitoramento, em que as informações de latitude e longitude da entidade (User2) são agrupadas com a resposta da pesquisa de metadados no banco de dados. Ao receber este objeto, a interface do usuário usa a posição do usuário (*position*) para movimentar o marcador no mapa, e o URI para exibir o vídeo da câmera da respectiva área de cobertura da entidade. O campo *cmd* é usado para informar quais tipos de informações estão sendo recebidos no cliente, por exemplo, caso o usuário esteja conectado e se movendo o comando recebido será *cmd=user-change* e a posição da entidade no mapa

é atualizado, mas se perder a conexão o comando é *cmd=desconectado* e o marcador do usuário do mapa é retirado.


```
{
  "user":"User2",
  "position":{
    "lat":-22.906382731234885,
    "lng":-43.13292718615037
  },
  "cmd":"user-change",
  "mongodb":[
    {
      "_id":"5c0eb8ecda54b426e9575eda",
      "user":"portal",
      "camname":"midiacom_2",
      "uri":"http://200.20.15.110:9004/video/5c0eb8ecda54b426e9575eda"
    }
  ]
}
```


Figura 6.3: Metadados enviados ao cliente (Interface do usuário)

A Figura 6.4 mostra o momento de funcionamento de uma consulta de metadados realizada diretamente pela interface do usuário ao módulo de consulta, e do servidor de vídeo. Quando se digita um endereço na caixa de texto, o mapa é posicionado para o endereço pesquisado e neste momento as coordenadas que delimitam a tela do mapa (canto superior direito e esquerdo, canto inferior direito e esquerdo) são enviados para a pesquisa de elementos nesta região da tela. Outro modo de se identificar os elementos no mapa é enquanto se movimentar sobre ele, em que novos parâmetros são colhidos do mapa para a realização de novas consultas. Na esquerda tem-se a localização da câmera, sua área de cobertura e uma fotografia que é tirada a cada minuto. Abaixo da imagem existe um link com o endereço IP do servidor de vídeo contendo a identificação da câmera e na direita a imagem da câmera sendo transmitida em tempo real.

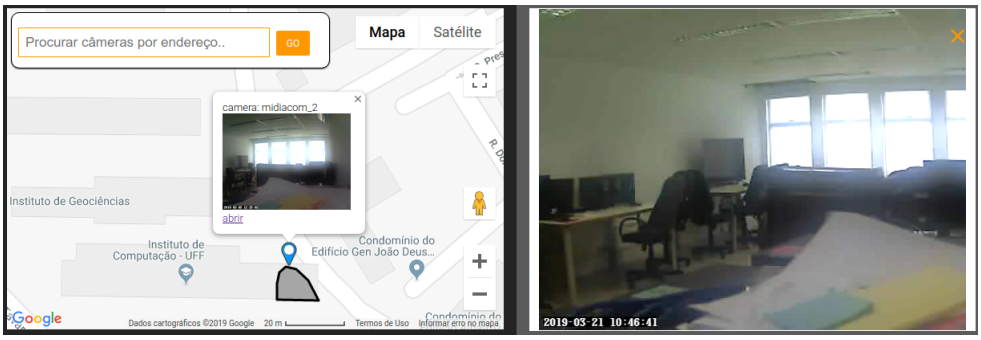


Figura 6.4: Funcionamento do servidor de vídeo desenvolvido

Uma consulta de elementos pode ser feita de dentro da interface Web do cliente ou

fora dela. Dentro da interface Web existe uma barra de localização para ser digitado um endereço ou mesmo através da navegaçao do mapa, em que as coordenadas retangulares da tela são enviadas automaticamente. Pela forma externa é possível usar uma URI, por exemplo, *http://dominio/lat/lng/r*, em que o primeiro parâmetro após o domínio é a latitude, o segundo é a longitude e o terceiro o raio de alcance em metros. Na Figura 6.5, é mostrado o retorno da consulta para latitude -22.906237, longitude -43.132918 e raio 50 metros. No campo *coordinates* existe o conjunto de coordenadas de cada ponto do polígono que representa a área de cobertura da câmera.


```
[
  {
    "_id": "5c0eb8ecda54b426e9575eda",
    "user": "portal",
    "camname": "midiacom_2",
    "position": {
      "type": "Point",
      "coordinates": [
        -43.13291353501961,
        -22.90635474895017
      ]
    },
    "uri": "http://200.20.x.x:9004/video/5c0eb8ecda54b426e9575eda",
    "public": "true",
    "coverarea": {
      "type": "Polygon",
      "coordinates": [
        [
          [
            -43.132953594015135,
            -22.906455719430035
          ],
          [
            -43.132953594015135,
            -22.90640630553734
          ],
          [
            -43.13294755904485,
            -22.906372333475655
          ],
          [
            -43.132924760268224,
            -22.906355077209252
          ],
          [
            -43.13288989155103,
            -22.90636310697108
          ],
          [
            -43.132850999520315,
            -22.90639769670886
          ],
          [
            -43.13281613080312,
            -22.906458228728788
          ],
          [
            -43.132953594015135,
            -22.906455719430035
          ]
        ]
      ]
    }
  }
]
```


Figura 6.5: Retorno de elemento através de consulta com HTTP

Na câmera utilizada existem um total de cinco parâmetros de qualidade (*Excellent, Detailed, Good, Standard, Medium*) para a transmisso do vídeo. Foram avaliados neste trabalho os parâmetros de maior qualidade (Excellent), o de qualidade média (Good) e o de baixa qualidade (Medium) em un período de observação de 2 minutos para cada parâmetro.

Nas Figuras 6.6 até 6.8 foram realizados o teste de consumo de banda no formato MJPEG diretamente da câmera sem a intervencão do servidor de vídeo, ou seja, as imagens foram transmitidas diretamente para a página HTML. Na Figura 6.6 com o parâmetro de melhor qualidade, a média de bits por segundo se manteve logo acima de 7,5Mbps na maior parte do tempo. Na Figura 6.7 com o parâmetro de qualidade mediana, o consumo médio foi entre 1,6Mbps e 1,75Mbps e, com parâmetro de pior qualidade, entre 0,9Mbps e 1Mbps na Figura 6.8. Na sequência da Figura 6.9 até 6.11, foi realizado o mesmo tipo de teste, mas agora utilizando a saída de fluxo em MP4 transcodificado para Theora no formato Ogg. Na Figura 6.9, a taxa média de bits ficou entre 300Kbps e 450bps. Na Figura 6.10, entre 300kbps e 320kbs e, na Figura 6.11, entre 220kbps e 280kbps.

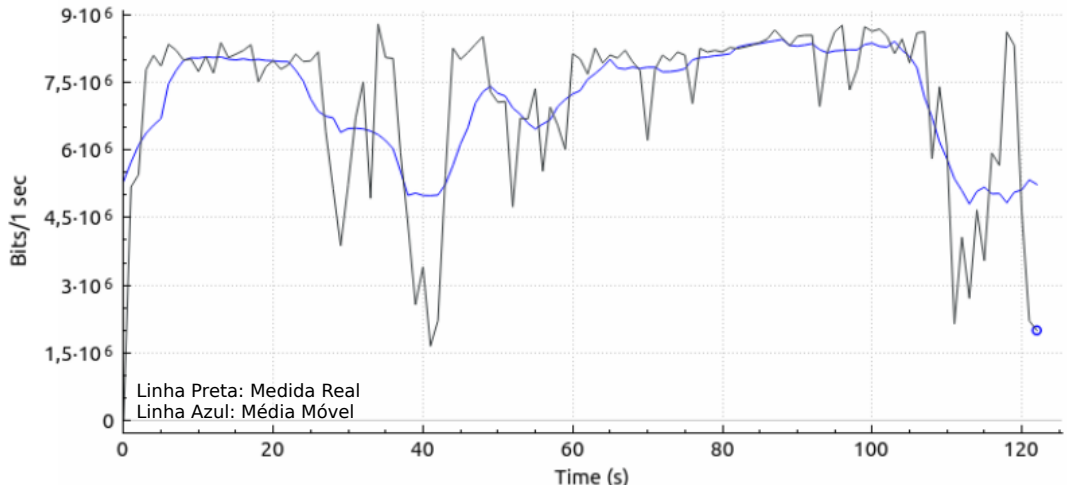


Figura 6.6: Parâmetro Excellent (Teste de consumo de banda para transmissão de vídeo no formato MJPEG diretamente para a página HTML).

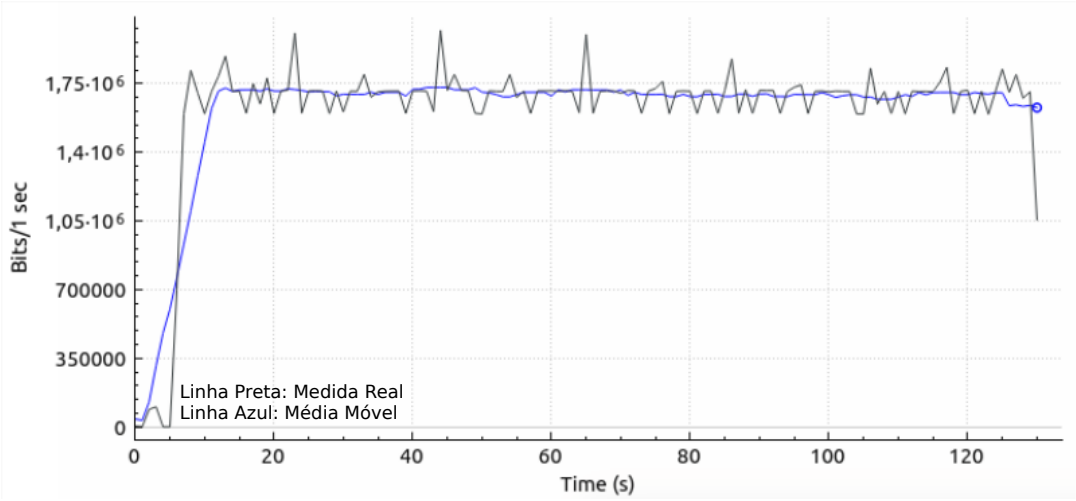
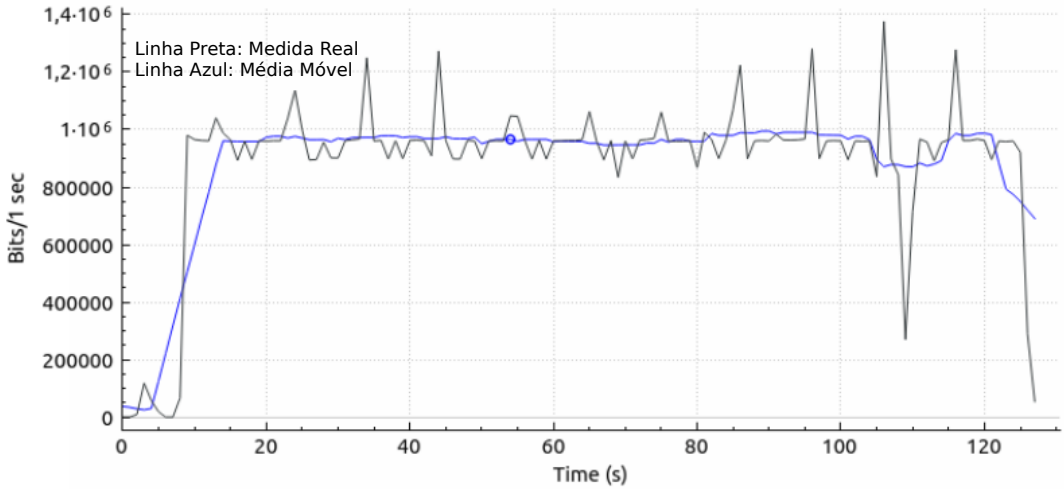


Figura 6.7: Parâmetro Good (Teste de consumo de banda para transmissão de vídeo no formato MJPEG diretamente para a página HTML).



(a)

Figura 6.8: Parâmetro Medium (Teste de consumo de banda para transmissão de vídeo no formato MJPEG diretamente para a página HTML).

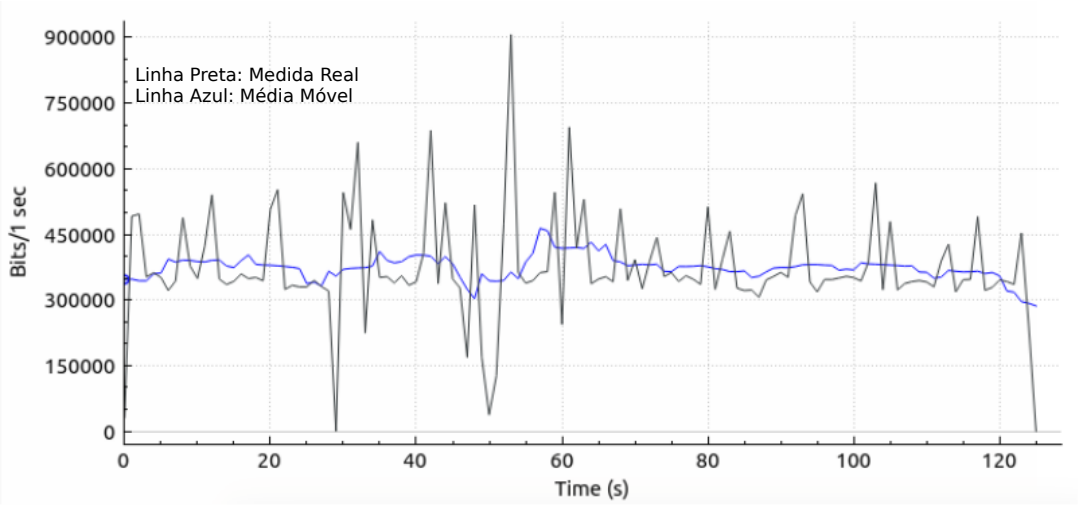


Figura 6.9: Parâmetro Excellent (Teste de consumo de banda para a transcodificação de vídeo no formato MP4 para Theora).

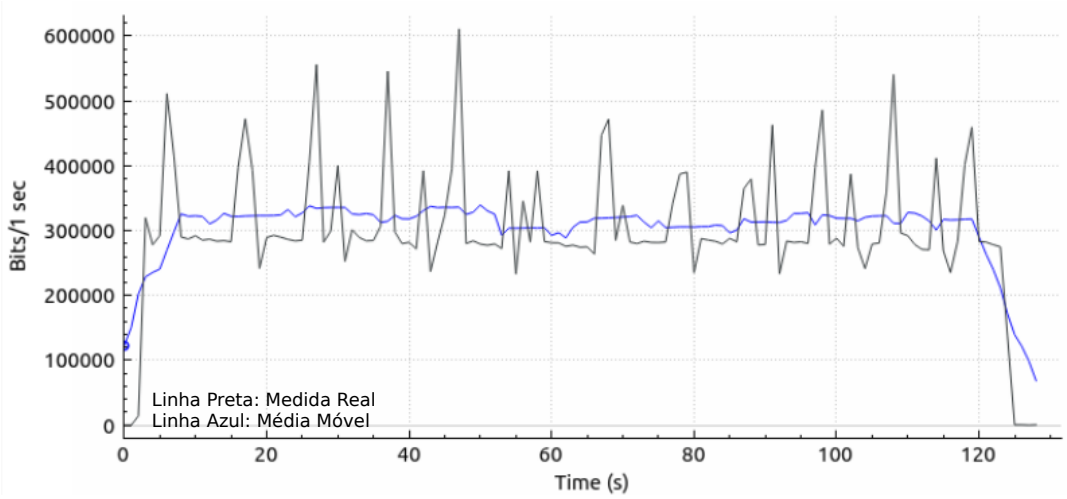


Figura 6.10: Parâmetro Good (Teste de consumo de banda para a transcodificação de vídeo no formato MP4 para Theora).

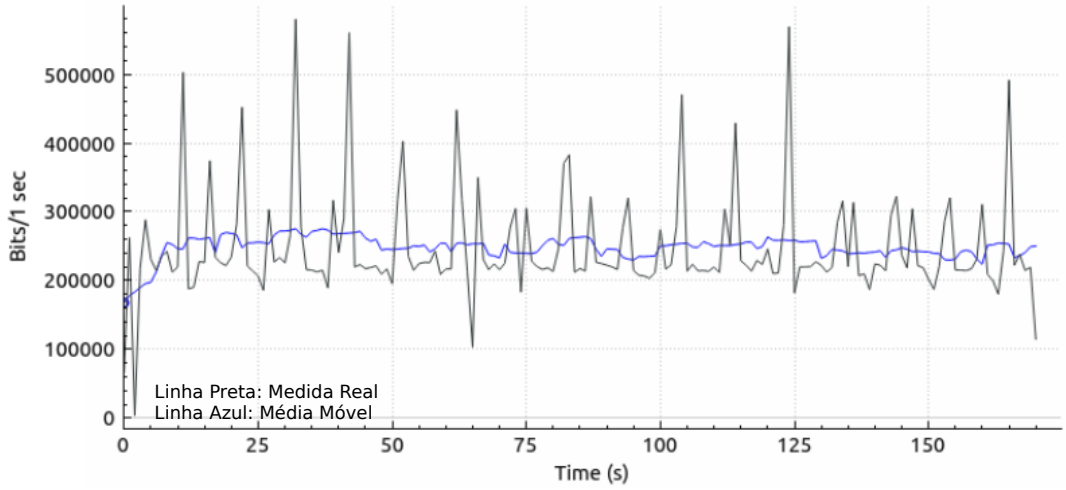


Figura 6.11: Parâmetro Medium (Teste de consumo de banda para a transcodificação de vídeo no formato MP4 para Theora).

Para a obtenção do atraso do vídeo foi feita a verificação de relógio em dois momentos. No primeiro momento, com a câmera configurada para registrar a hora da filmagem nas próprias imagens, é feito a sincronização com um servidor NTP (*Network Time Protocol*). No segundo momento, com a máquina cliente também sincronizada com o servidor NTP, são anotadas as diferenças dos horários da máquina local e do horário contido nas imagens. A diferença entre a hora da marcação da câmera e da apresentação no navegador é considerado o atraso de transmissão do vídeo pela rede. Os resultados obtidos estão organizados nas Tabelas 6.5 e 6.6 da Seção 6.2

A configuração de hardware dos equipamentos utilizados estão nas Tabelas 6.1, 6.2, 6.3 e 6.4.

Tabela 6.1: Configuração do Servidor Web e Servidor de Monitoramento (Mesma máquina)

Item	**Descrição**
Memória	8GB
Processoador	Intel® Core™ i7 2.67GHz × 8
Sistema	Linux 64-bit

Tabela 6.2: Configuração do Servidor de Vídeo (Máquina Virtual)

Item	**Descrição**
Memória	2 GB
Processoador	Intel® Xeon™ CPU E5-2650 2.0GHz
Sistema	64-bit

Tabela 6.3: Configuração da máquina Cliente

Item	**Descrição**
Memória	4 GB
Processoador	Intel® Core™ i5 2.50GHz × 4
Sistema	Linux 64-bit

Tabela 6.4: Configuração da Câmera IP

Item	**Descrição**
Modelo	AirCam WL-350HD
Image sensor	1/4" Progressive scan CMOS 1.3 Megapixel sensor
Lens	F1.8, 4.2 mm
Digital Zoom	10x digital
Min Illumination	0 Lux at F2.0 with Night-vision LEDs on
IR working distance	5 M
Video Compression	Motion JPEG MPEG-4 Part2 (ISO/IEC 14496-2) Profile: SP H.264 BP

6.2 Análise dos Resultados

Na Tabela 6.5 estão mostrados os resultados do teste para a transferência de vídeo escolhendo 3 parâmetros de qualidade existentes na câmera (*Excellent, Good, Medium*). Como pode-se observar, o método de codificação MJPEG (Motion JPEG) utiliza uma taxa de bits por segundo bastante considerável em todas as categorias. O vídeo em MJPEG pode ser acessado diretamente dentro de uma página Web sem a necessidade de Plugin, e esse é um ponto positivo para transferir o vídeo sem precisar ser transcodificado. Contudo, o uso intenso com altas taxas de bits dificulta a escalabilidade do serviço sendo necessário usar métodos mais avançados de codificação. Para o método de codificação Theora, os resultados foram melhores, mesmo para o caso de melhor qualidade. Apesar de neste caso ser necessário fazer uma conversão de protocolos de RSTP para HTTP e uma transcodificação de vídeo, o ganho em largura de banda indica que este caminho é o mais adequado para que se possa utilizar várias câmeras.

Tabela 6.5: Média de bits por segundo para parâmetros de qualidade

bps médio	Excellent	Good	Medium
MJPEG	[7,5 - 8,5]Mbps	[1,6 - 1,75]Mbps	[0,9 - 1]Mbps
Theora	[300 - 400]Kbps	[300 - 320]Kbps	[200 - 280]Kpbs

Na Tabela 6.6, estão registrados os tempos gastos ou atrasos para que o vídeo pro-

duzido pela câmera chegue ao cliente. Este teste precisou ser realizado para validar tecnicamente a arquitetura proposta na Internet onde existem flutuações do tráfego da rede. Com a utilização do método HLS (HTTP *Live Streaming*), houve um tempo maior devido à criação de pequenos arquivos que são transmitidos para a página web. O servidor de vídeo desenvolvido não gera arquivos, ele aguarda uma requisão para transmitir os últimos bits que chegaram da câmera para serem enviados. Desta maneira, o tempo levado para que o vídeo seja apresentado na página Web mostrou que o método empregado é mais eficiente, permitindo o monitoramento com um atraso baixo.

Tabela 6.6: Atrasos na recepção das imagens, n=10 (amostras)

Método	Atraso médio	Intervalo de confiança de 95%
Nginx/HLS	34,2s	[32,54s - 35,86s]
SVD/Theora	6,1s	[5,64s - 6,56s]

Capítulo 7

Conclusão

As contribuições deste trabalho foram propor uma arquitetura georreferenciada (baseada em localização) para seleção de fluxo de vídeo para o monitoramento de entidades, a elaboração de um modelo para a representação de áreas de cobertura de câmeras de monitoramento, a definição de uma abordagem dinâmica para o armazenamento e consulta de informações sobre as câmeras e a implementação de um protótipo para a avaliação da arquitetura proposta, bem como a verificação de viabilidade de realizar o monitoramento em tempo real na Internet sobre condições variadas de tráfego. Além disso foi desenvolvido um servidor de vídeo para a Web.

A definição para o modelo de metadados para a representação das áreas de coberturas baseou-se em associar coordenadas geográficas para os vértices da figura geométrica fechada. Após o conceito definido foi feita uma pesquisa sobre sistemas GIS, Web mapping e de banco de dados que pudessem trabalhar com modelos de dados geográficos. O teste compreendeu um cenário utilizando uma câmera IP, um servidor de vídeo e uma entidade enviando a sua posição GPS para o módulo de monitoramento e visualizado pela interface de usuário. Além do teste de monitoramento, também foi realizada a busca de elementos de fora da interface do usuário (Web), usando apenas uma URI contendo uma localização geográfica e um raio de cobertura possibilitando que outros sistemas acessem os metadados das câmeras. Verificou-se também a banda consumida para a transmissão do vídeo e o tempo gasto até a sua apresentação no terminal do cliente.

A avaliação de consumo de banda se deu sobre a codificação MJPEG e Theora. O MJPEG foi testado devido à câmera já dispor de uma URL com acesso direto ao vídeo compatível com o navegador Web sem a necessidade de conversão usando o protocolo HTTP, contudo, o consumo de banda foi excessivo tornando sua utilização inviável para a escalabilidade do sistema. A codificação Theora foi aplicada após a conversão do protocolo

RTSP, que é usado pela câmera para o protocolo HTTP, sendo o formato de vídeo Theora compatível com os navegadores Chrome e Theora. O consumo de banda desta codificação apresentou melhoras significativas com relação à codificação MJPEG.

Para avaliar o atraso do vídeo desde a sua geração na câmera até a apresentação no equipamento cliente os relógios da câmera e do cliente foram sincronizados com um servidor NTP e para a transferência do vídeo foram realizados testes com as técnicas HLS (HTTP *Live Streaming*) e um modelo desenvolvido de servidor de vídeo para transmissão rápida de imagens. O atraso médio de 34,2s pela técnica HLS foi devido à criação de pequenos arquivos de vídeo e um índice contendo metadados destes arquivos, sendo que no servidor de vídeo desenvolvido o atraso foi de 6,1s devido aos bits serem codificados e enviados diretamente para o navegador Web.

Foi demonstrada a eficácia da proposta do monitoramento com baixa latência na transmissão de vídeo, mas ainda existem limitações. As limitações do monitoramento compreendem a falta de um teste de escala utilizando muitas entidades com várias câmeras simultaneamente. A pouca acurácia das localizações de GPS e a falta de compatibilidade do formato Ogg/Theora para outros navegadores.

7.1 Trabalhos Futuros

Para trabalhos futuros, planejam-se melhorias para permitir o uso de codificações compatíveis com todos os navegadores e diminuição dos atrasos de vídeo, o uso de predição de localização para evitar gastos de bateria do sistema móvel incluindo técnicas de computação visual para a identificação de entidades para os casos em que não se usa GPS.

Uma segunda etapa de pesquisa poderá ser a gravação de arquivos de vídeo e dos registros dos trajetos das entidades. Neste caso a proposta é voltada para eventos passados, em que possam ser resgatados os trechos dos vídeos referentes ao intervalo de tempo em que a entidade transitou sobre a área de cobertura.

O desenvolvimento de um servidor de vídeo abre caminho para outros tipos serviços além do monitoramento. O desenvolvimento de *streaming* com baixa latência e com baixas taxas de transmissão pode permitir a criação de um portal público colaborativo em que as pessoas podem cadastrar suas câmeras ajudando a segurança em um bairro ou cidade. Para isso, um sistema de gravação de arquivos de vídeo em nuvem pode ser um tema de pesquisa.

O banco de dados de metadados de câmeras também pode ter seu conceito expandido a outros tipos de equipamentos, bem como o desenvolvimento de APIs para que os cadastros sejam feitos fora do portal contribuindo para a expansão e descoberta de novos itens.

Referências

[1] Alvi, S. A.; Afzal, B.; Shah, G. A.; Atzori, L.; Mahmood, W. Internet of multimedia things: Vision and challenges. *Ad Hoc Networks 33* (2015), 87–111.

[2] Anagnostopoulos, C. N. E.; Anagnostopoulos, I. E.; Loumos, V.; Kayafas, E. A license plate-recognition algorithm for intelligent transportation system applications. *IEEE Transactions on Intelligent transportation systems 7*, 3 (2006), 377–392.

[3] Apple. Http live streaming. Tech. rep., August 2017. `https://tools.ietf.org/html/rfc8216`.

[4] Atzori, L.; Iera, A.; Morabito, G. The internet of things: A survey. *Computer networks 54*, 15 (2010), 2787–2805.

[5] Benenson, R.; Omran, M.; Hosang, J.; Schiele, B. Ten years of pedestrian detection, what have we learned? In *European Conference on Computer Vision* (2014), Springer, pp. 613–627.

[6] Chang, K.-T. *Introduction to geographic information systems*. McGraw-Hill Higher Education, 2019.

[7] Chang, S.-L.; Chung, Y.-C.; Chen, S.-W. Automatic license plate recognition. ieee transactions on intelligent transportation systems.

[8] Chen, Y.-L.; Chen, T.-S.; Huang, T.-W.; Yin, L.-C.; Wang, S.-Y.; Chiueh, T.-c. Intelligent urban video surveillance system for automatic vehicle detection and tracking in clouds. In *2013 IEEE 27th international conference on advanced information networking and applications (AINA)* (2013), IEEE, pp. 814–821.

[9] Chenshu Wu, Zheng Yang, Y. X. Y. Z. Y. L. Human mobility enhances global positioning accuracy for mobile phone localization. *IEEE Transactions on Parallel and Distributed Systems 26*, 3 (2015), 131–141.

[10] Chodorow, K. *MongoDB: The Definitive Guide: Powerful and Scalable Data Storage*. "O'Reilly Media, Inc.", 2013.

[11] Chris Kiefer, F. B. Smart e-bike monitoring system: real-time open source and open hardware gps assistance and sensor data for electrically-assisted bicycles. *IET Intelligent Transport Systems 10*, 2 (2016), 79–88.

[12] Cisco. The zettabyte era: Trends and analysis. Tech. rep., June 2017. `https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html`.

[13] Dey, M.; Arif, M. A.; Mahmud, M. A. Anti-theft protection of vehicle by gsm amp; gps with fingerprint verification. In *2017 International Conference on Electrical, Computer and Communication Engineering (ECCE)* (Feb 2017), pp. 916–920.

[14] (FCC), F. C. C. *911 and E911 Services.* `https://www.fcc.gov/general/9-1-1-and-e9-1-1-services`.

[15] Ferrando, S.; Gera, G.; Regazzoni, C. Classification of unattended and stolen objects in video-surveillance system. In *2006 IEEE International Conference on Video and Signal Based Surveillance* (2006), IEEE, pp. 21–21.

[16] Fette, I.; Melnikov, A. The websocket protocol. Tech. rep., 2011.

[17] Force, I. E. T. Rtp: A transport protocol for real-time applications. Tech. rep., 2003. `https://tools.ietf.org/html/rfc3550`.

[18] Gama, K.; Touseau, L.; Donsez, D. Combining heterogeneous service technologies for building an internet of things middleware. *Computer Communications 35*, 4 (2012), 405–417.

[19] Gibbons, P. B.; Karp, B.; Ke, Y.; Nath, S.; Seshan, S. Irisnet: An architecture for a worldwide sensor web. *IEEE pervasive computing 2*, 4 (2003), 22–33.

[20] Goldberg, D. W.; Wilson, J. P.; Knoblock, C. A. From text to geographic coordinates: the current state of geocoding. *URISA journal 19*, 1 (2007), 33–47.

[21] Green, M. W. The appropriate and effective use of security technologies in u.s. schools. a guide for schools and law enforcement agencies. *Sandia National Laboratories* (1999).

[22] Guinard, D.; Trifa, V. Towards the web of things: Web mashups for embedded devices. In *Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences), Madrid, Spain* (2009), vol. 15.

[23] Han, J.; Haihong, E.; Le, G.; Du, J. Survey on nosql database. In *Pervasive computing and applications (ICPCA), 2011 6th international conference on* (2011), IEEE, pp. 363–366.

[24] Hassan, M. M.; Hossain, M. A.; Al-Qurishi, M. Cloud-based mobile iptv terminal for video surveillance. In *16th International Conference on Advanced Communication Technology* (2014), IEEE, pp. 876–880.

[25] Hui-Lee Ooi, Guillaume-Alexandre Bilodeau, N. S. D.-A. B. Multiple object tracking in urban traffic scenes with a multiclass object detector. In *International Symposium on Visual Computing* (2018), Springer, pp. 727–736.

[26] Iliffe, J. *Datums and map projections for remote sensing, GIS, and surveying.* CRC Press, 2000.

[27] Ionut Constandache, Romit Roy Choudhury, I. R. Towards mobile phone localization without war-driving. *2010 Proceedings IEEE INFOCOM* (2010), 1–9.

[28] Jenkins, N. 245 million video surveillance cameras installed globally in 2014, June 11 2015.

[29] Krumm, J. *Ubiquitous computing fundamentals.* Chapman and Hall/CRC, 2016.

[30] Kurose, J.; Ross, K. *Computer Networking: A Top-Down Approach.* Pearson Education, 2013.

[31] Lavee, G.; Khan, L.; Thuraisingham, B. A framework for a video analysis tool for suspicious event detection. *Multimedia Tools and Applications 35*, 1 (2007), 109–123.

[32] Lei, K.; Ma, Y.; Tan, Z. Performance comparison and evaluation of web development technologies in php, python, and node. js. In *2014 IEEE 17th international conference on computational science and engineering* (2014), IEEE, pp. 661–668.

[33] Lenz, R. K.; Tsai, R. Y. Techniques for calibration of the scale factor and image center for high accuracy 3-d machine vision metrology. *IEEE Transactions on pattern analysis and machine intelligence 10*, 5 (1988), 713–720.

[34] Liao, H.-C.; Chu, P.-T. A novel visual tracking approach incorporating global positioning system in a ubiquitous camera environment. *Information Technology Journal 8*, 4 (2009), 465–475.

[35] Maggio, E.; Cavallaro, A. *Video tracking: theory and practice.* John Wiley & Sons, 2011.

[36] MANIKANDAN, R.; Niranjani, S. Implementation on real time public transportation information using gsm query response system. *Contemporary Engineering Sciences 7* (05 2014), 509 – 514.

[37] MongoDb. `https://docs.mongodb.com/manual/`.

[38] Nam, Y.; Rho, S.; Park, J. H. Intelligent video surveillance system: 3-tier context-aware surveillance system with metadata. *Multimedia Tools and Applications 57*, 2 (2012), 315–334.

[39] Neha Mangla, G Sivananda, A. K. V. A gps-gsm predicated vehicle tracking system, monitored in a mobile app based on google maps. *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)* (2017), 2916–2919.

[40] Neis, P.; Zielstra, D. Recent developments and future trends in volunteered geographic information research: The case of openstreetmap. *Future Internet 6*, 1 (2014), 76–106.

[41] Netgear. Ip networking and its impact on video surveillance. Tech. rep., 2012. `https://www.netgear.com/images/ip%20networking%20and%20its%20impact%20on%20video%20surveillance18-51824.pdf`.

[42] Nunamaker Jr, J. F.; Chen, M.; Purdin, T. D. Systems development in information systems research. *Journal of management information systems 7*, 3 (1990), 89–106.

[43] Qian (Chayn) Sun, Robert Odolinski, J. C. X. J. F. T. F. H. L. Validating the efficacy of gps tracking vehicle movement for driving behaviour assessment. *Travel Behaviour and Society 6* (2017), 32–43.

[44] R.Maruthi, C. Sms based bus tracking system using open source technologies. *International Journal of Computer Applications 86*, 9 (02 2014).

[45] Shunsuke Miura, Li-Ta Hsu, F. C. S. K. Gps error correction with pseudo-range evaluation using three-dimensional maps. *IEEE Transactions on Intelligent Transportation Systems 16*, 6 (2015), 3104–3115.

[46] Sodagar, I. The mpeg-dash standard for multimedia streaming over the internet. *IEEE MultiMedia 18*, 4 (2011), 62–67.

[47] Son, J.; Kim, S.; Sohn, K. Fast illumination-robust foreground detection using hierarchical distribution map for real-time video surveillance system. *Expert Systems with Applications 66* (2016), 32–41.

[48] Szwoch, G. Extraction of stable foreground image regions for unattended luggage detection. *Multimedia Tools and Applications 75*, 2 (2016), 761–786.

[49] Tilkov, S.; Vinoski, S. Node. js: Using javascript to build high-performance network programs. *IEEE Internet Computing 14*, 6 (2010), 80–83.

[50] Valera, M.; Velastin, S. A. Intelligent distributed surveillance systems: a review. *IEE Proceedings-Vision, Image and Signal Processing 152*, 2 (2005), 192–204.

[51] Wang, W.; Gee, T.; Price, J.; Qi, H. Real time multi-vehicle tracking and counting at intersections from a fisheye camera. In *2015 IEEE Winter Conference on Applications of Computer Vision* (2015), IEEE, pp. 17–24.

[52] Xia, F.; Yang, L. T.; Wang, L.; Vinel, A. Internet of things. *International Journal of Communication Systems 25*, 9 (2012), 1101–1102.

[53] Xiaoming, L.; Tian, Q.; Wanchun, C.; Xingliang, Y. Real-time distance measurement using a modified camera. In *Sensors Applications Symposium (SAS), 2010 IEEE* (2010), IEEE, pp. 54–58.

[54] Xin, J.-n.; Du, X.; Zhang, J. Deep learning for robust outdoor vehicle visual tracking. In *2017 IEEE International Conference on Multimedia and Expo (ICME)* (2017), IEEE, pp. 613–618.

[55] Yilmaz, A.; Javed, O.; Shah, M. Object tracking: A survey. *Acm computing surveys (CSUR) 38*, 4 (2006), 13.