ALGORITMO PARA LA ESTIMACIÓN DE ALTURAS DE EDIFICIOS EN IMÁGENES DE GOOGLE STREET-VIEW USANDO METROLOGÍA DE UNA VISTA

ELKIN DAVID DÍAZ PLATA

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERIAS FISICO-MECANICAS

ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMATICA

BUCARAMANGA

2016

ALGORITMO PARA LA ESTIMACIÓN DE ALTURAS DE EDIFICIOS EN IMÁGENES DE GOOGLE STREET-VIEW USANDO METROLOGÍA DE UNA VISTA

ELKIN DAVID DÍAZ PLATA

Trabajo de grado para optar al título de

Ingeniero de sistemas

Director:

PhD. HENRY ARGUELLO FUENTES

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERIAS FISICO-MECANICAS

ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMATICA

BUCARAMANGA

2016

La ciencia será siempre una búsqueda, jamás un descubrimiento real; Es un viaje, nunca una llegada.

Karl Popper

A mi madre, por darme la vida y darme su infinito amor, apoyo y confianza.

A mi compañera de vida, Silvia, por su cariño y darme motivos para seguir luchando.

A mis hermanos, por brindarme valiosas lecciones de vida.

A mis amigos más cercanos, en especial Pedro, por su compañía y darme un lugar en quien confiar plenamente.

A mi maestro y director, Henry Arguello, por darme a conocer una filosofía diferente, sus grandes enseñanzas y por guiarme en el camino de la ciencia.

ÍNDICE GENERAL

INTRODUCCIÓN 12
1. CONCEPTOS BÁSICOS 13
1.1 Google Street-View 13
1.1.1 Google Static Maps API 14
1.1.2 Google Maps Roads API 15
1.1.3 Google Street View Image API 16
1.2 Transferencia de Estado Representacional (REST) 17
1.3 REST aplicado a HTTP 21
1.4 Geometría proyectiva 21
1.5 Metrología de una vista 23
2. MÉTODOS TRADICIONALES DE ESTIMACIÓN DE ALTURAS EN EDIFICIOS. 26
2.1 Métodos basados en la sombra de los edificios 26
2.2 Métodos basados en telémetros y sensores LiDAR 26
2.3 Métodos basados en imágenes SAR (Synthetic aperture radar) 27
3. MÉTODO PROPUESTO USANDO IMÁGENES DE STREET-VIEW Y METROLOGÍA DE UNA VISTA 28
3.1 Sistema de transferencia de estado representacional (REST) 28
3.2 GeoJSON 30
3.3 Metrología de una vista 30
4. Resultados 34
5. Conclusiones 40
Referencias 41
Bibliografía 45

ÍNDICE DE FIGURAS

Figura 1 Ejemplo de Respuesta de la Google Maps Static API 15
Figura 2 Parámetros pitch y heading para la misma posición (latitud y longitud) 16
Figura 3 Estilo Cliente-Servidor 17
Figura 4 Estilo Cliente-sinEstado-Servidor 18
Figura 5 Estilo Cliente-Caché-sinEstado-Servidor 19
Figura 6 Estilo Uniforme-Cliente-Caché-sinEstado-Servidor 20
Figura 7 Transferencia de Estado Representacional (REST) 21
Figura 8 Proyección de un punto en el espacio 3D a un punto 2D en el plano de la imagen 22
Figura 9 Geometría básica 23
Figura 10 Horizonte o línea de fuga y puntos de fuga 23
Figura 11 Distancia entre dos planos 24
Figura 12 Diagrama de flujo del algoritmo propuesto 29
Figura 13 Área de la ciudad de Bucaramanga 34
Figura 14 Resultado de la estimación para el área norte de Bucaramanga 35
Figura 15 Visualización de la capa de alturas sobre Google Maps en Bucaramanga 35
Figura 16 Área de la ciudad de Madrid 37
Figura 17 Resultado de la estimación para el área de la ciudad de Madrid 37
Figura 18 Visualización de la capa de alturas sobre Google Maps en Madrid 38
Figura 19 Altura estimada por el algoritmo vs Altura estimada real 39

ÍNDICE DE TABLAS

Tabla 1 Parámetros más importantes en Google Static Maps API 14
Tabla 2 Parámetros más importantes en Google Street View Image API 16
Tabla 3 Resultados experimentales para el área de Bucaramanga 36

Resumen

TITULO: ALGORITMO PARA LA ESTIMACIÓN DE ALTURAS DE EDIFICIOS EN IMÁGENES DE GOOGLE STREET-VIEW USANDO METROLOGÍA DE UNA VISTA[1]

AUTOR: ELKIN DAVID DÍAZ PLATA[2]

PALABRAS CLAVE: Google Street-View, Imágenes públicas, Procesamiento de imágenes, Geometría proyectiva, REST, Modelo de cámara estenopeica, Altura de edificios.

DESCRIPCIÓN

La detección y el análisis de edificios es una tarea fundamental en el monitoreo urbano, el control y la planificación. La extracción de la altura de edificios y su densidad es un aspecto importante en las investigaciones de teledetección urbana, estas estimaciones proporcionan una base científica para la planificación de las ciudades y la construcción ecológica urbana. Con el fin de lograr estos objetivos es necesario tener algunos datos como la altura de los edificios y su densidad.

Los métodos tradicionales de estimación de alturas se basan en dispositivos de hardware especial y son de alto costo monetario. La mayoría de las ciudades importantes del mundo han sido fotografiadas por Google Street-View en diferentes condiciones. Estas imágenes pueden ser usadas para estimar la altura de los edificios a un costo reducido. En los últimos años se han desarrollado técnicas de medición en imágenes de una sola vista sin tener en cuenta los parámetros intrínsecos de la cámara. Estos avances han facilitado el proceso de medición en imágenes en donde se desconocen los parámetros de la cámara con la cual fue tomada. Aplicar metrología de una vista en la estimación de alturas de edificios en imágenes de Google Street-View, disminuyendo el costo de obtención de esta medida, es la principal motivación de este proyecto. Específicamente, el objetivo de este trabajo es desarrollar un algoritmo computacional para obtener de forma automática miles de imágenes georreferenciadas de Google Street-View a través de la determinación de un sistema de transferencia de estado representacional y estimar su altura media usando metrología de una sola vista. Además, las mediciones resultantes y los metadatos de las imágenes serán utilizadas para derivar una capa de alturas en un mapa de Google disponible en la web.

[1] Trabajo de investigación

[2] Facultad de ingenierías Fisicomecánicas. Escuela de ingeniería de Sistemas e Informática. Director, Profesor Henry Arguello Fuentes

ABSTRACT

TITLE: ALGORITHM FOR ESTIMATING HEIGHTS OF BUILDINGS IN IMAGES OF GOOGLE STREET-VIEW USING SINGLE VIEW METROLOGY[3]

AUTHOR: ELKIN DAVID DÍAZ PLATA[4]

KEYWORDS: Google Street-View, Public images, Image processing, projective geometry, REST, pinhole camera model, height of buildings.

DESCRIPTION:

The detection and analysis of buildings is a key task in urban monitoring, control and planning. The extraction and density of buildings is an important research aspect of urban remote sensing, these estimates provide a scientific basis for the planning of cities and urban ecological construction. In order to achieve these objectives it is necessary to have some data such as building height and density.

Traditional methods of estimating heights are based on special hardware devices and are of high monetary cost. Most major world cities have been photographed by Google Street-View in different conditions. These images can be used to estimate the height of the buildings at a reduced cost. In recent years techniques have been developed measuring in images from a single view regardless of the intrinsic parameters of the camera. These advances have facilitated the measurement process images where the parameters of the camera that was taken is unknown. Apply Single View Metrology in estimating heights of buildings in Google StreetView images, reducing the cost of obtaining this measure, it is the main motivation for this project. Specifically, the aim of this work is to develop a computational algorithm for automatically obtain thousands of geolocated images Google Street-View through the determination of a system of representational state transfer and estimate their average height using single view metrology. In addition, the resulting measurements and metadata of the images will be used to derive a layer heights in a Google Map available on the web.

[3] Research Work

[4] School of Physical-Mechanical Engineering. Department of Systems Engineering and Informatics. Advisor, Professor Henry Arguello Fuentes

INTRODUCCIÓN

La proliferación de servicios gratuitos en internet en las últimas décadas ha permitido que empresas como Google suministren grandes cantidades de información digital de abundante riqueza. Google *Street-view* desarrollado por Google Inc, es un servicio que proporciona imágenes panorámicas a nivel de calle (360 grados de movimiento horizontal y 290 grados de movimiento vertical), que permite obtener imágenes de una ciudad casi en su totalidad. La disponibilidad de este conjunto de imágenes y similares (*StreetSide* de Microsoft) representa enormes oportunidades en el campo visión por computador, donde los algoritmos podrían analizar y obtener información valiosa de una ciudad sin incurrir en gastos elevados. Específicamente, la altura de los edificios es un parámetro de gran utilidad que puede ser extraído por dichos algoritmos a partir de éstas imágenes.

La extracción de la altura de edificios es un aspecto importante en las investigaciones de monitoreo urbano, el control y la planificación. Sin embargo, los métodos tradicionales para extraer esta información dependen, en gran medida, de la iluminación del sol y un tiempo libre de nubes. Además, generalmente son muy costosos y requieren de hardware especializado como los radares de apertura sintética, que proporcionan imágenes independientemente del día y de las condiciones meteorológicas en el momento de la adquisición.

Las imágenes de Google *Street-View* han sido utilizadas en trabajos anteriores, incluyendo la auditoría de entornos vecinales [1] y de entorno construido [2], la detección de peatones [3], posicionamiento 3D [4], modelamiento 3D [5], extracción de señales de tránsito [6] y segmentación [7]. Sin embargo, ningún trabajo ha utilizado estas imágenes para hacer mediciones. Además en estos trabajos se utiliza una mínima parte del conjunto de imágenes de Google.

Utilizando un sistema de transferencia de estado representacional es posible obtener todas las imágenes de Google *Street-View* de cualquier ciudad para luego estimar las alturas de los edificios con la técnica de metrología de una vista.

El desarrollo de técnicas de medición en imágenes de una sola vista, sin tener en cuenta los parámetros intrínsecos de la cámara, ha facilitado el proceso de medición en imágenes en las que se desconocen los parámetros del sensor. El uso de la metrología de una vista para estimar distancias en una imagen permite aprovechar la geometría proyectiva, presente en las imágenes, para realizar una medida exacta a partir de una referencia o la altura de la cámara [8].

Este trabajo está enfocado en el desarrollo de un sistema de transferencia de estado representacional, que permite obtener las imágenes de Google *Street-View,* para luego realizar la estimación de alturas de edificaciones usando metrología de una vista mediante un algoritmo propuesto. Adicionalmente, el algoritmo crea una capa de alturas que puede ser leída por Google *Maps* para ser mostrada en la web.

Capítulo 1

CONCEPTOS BÁSICOS

1.1 Google Street-View

Google Street View es una tecnología destacada de Google Maps y Google Earth que ofrece vistas panorámicas de imágenes pegadas de las principales calles de casi todas las ciudades del mundo [9]. Estas imágenes son tomadas desde 9 cámaras montadas sobre una flota de automóviles. Desde su lanzamiento, se han usado 4 tipos de cámaras diferentes. Las primeras tres generaciones de cámaras tomaban imágenes de baja calidad. La cuarta generación de cámaras corresponde a cámaras con capacidad de tomar imágenes en alta definición.

En primer lugar, se circula con los vehículos y se fotografían los lugares que se van a mostrar en Street-View. Se tienen en cuenta numerosos factores, como las condiciones atmosféricas y la densidad de población de diversas zonas, para determinar cuándo y dónde es posible obtener las mejores imágenes. Para relacionar cada imagen con su ubicación geográfica en el mapa, se combinan las señales de los sensores del vehículo que estima la posición, la velocidad y la dirección. Esto ayuda a reconstruir la ruta exacta del vehículo e incluso a inclinar y realinear las imágenes según sea necesario. Para evitar espacios en las panorámicas, unas cámaras adyacentes adquieren imágenes ligeramente superpuestas, que luego se pegan para crear una única imagen de 360 grados. Después, se aplican algoritmos de procesamiento de imágenes para reducir las uniones y crear transiciones suaves.

La mayoría de las fotografías se hacen en vehículo, pero algunas son tomadas por excursionistas, caminante, en barco, en moto de nieve, en camello y en buzos. Google Street View está disponible como un componente de Google Maps, como un aplicativo web y como una aplicación móvil para Android e iOS.

Google ofrece un conjunto de APIs [10], que incluyen rutinas, protocolos y herramientas para desarrollar aplicaciones software que permiten acceder a los servicios y datos de sus servidores con restricciones usando REST.

1.1.1 Google Static Maps API

El API de Google *Static Maps* devuelve una imagen (ya sea GIF, PNG o JPEG) en respuesta a una petición HTTP a través de una URL. Para cada solicitud, se puede especificar la ubicación del mapa, el tamaño de la imagen, el nivel de amplificación, el tipo de mapa y la colocación de marcadores opcionales en lugares en el mapa. Se puede etiquetar adicionalmente sus marcadores utilizando caracteres alfanuméricos [11]. La figura 1.1 muestra un ejemplo de una respuesta de Google *Static Maps* API.

Una URL se utiliza siguiendo el siguiente formato:

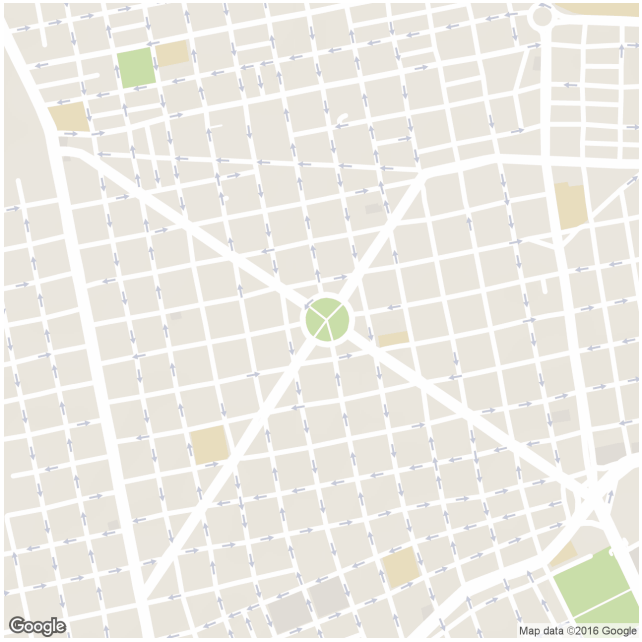
```
https://maps.googleapis.com/maps/api/staticmap?parameters
```

Algunos parámetros de URL son obligatorios, mientras que algunos son opcionales. Como es norma en las URL, todos los parámetros se separan mediante el carácter "&". La lista de parámetros más importantes y sus valores posibles se muestran a continuación.

Tabla 1: Parámetros más importantes en Google Static Maps API

Parámetros más importantes en Google Static Maps API	
Parámetro	**Descripción**
center	Define el centro del mapa usando longitud y latitud o una dirección.
zoom	Define el nivel de acercamiento del mapa usando un valor numérico.
size	Define las dimensiones rectangulares de la imagen del mapa usando valores numéricos (valor_horizontal x valor_vertical).
scale	Define la cantidad de pixeles que contiene la imagen devuelta.
maptype	Define el tipo de mapa a mostrar (por ejemplo, *terrain.*).
style	Define un estilo personalizado para alterar la presentación de una característica específica (carreteras, parques, etc.) del mapa.

Figura 1. Ejemplo de Respuesta de la Google Maps Static API a las url [12]



1.1.2 Google Maps Roads API

El API de caminos de mapas de Google permite mapear coordenadas GPS a la geometría del camino [13]. El servicio snap to roads devuelve el mejor ajuste de la geometría del camino para un conjunto de coordenadas GPS. Este servicio toma hasta 100 coordenadas GPS obtenidas a lo largo de un camino y devuelve un conjunto similar de datos con las coordenadas ajustadas al camino más probable que un vehículo pueda circular.

Una URL se utiliza siguiendo el siguiente formato:

```
https://roads.googleapis.com/v1/snapToRoads?parameter
```

Entre los parámetros más importantes se encuentra *path* el cual acepta una lista de pares latitud/longitud.

1.1.3 Google Street View Image API

El API de imágenes de *Street View* de Google permite obtener un panorama de Street View sin necesidad de usar JavaScript o alguna carga dinámica de contenido [14]. Este servicio crea una imagen basado en parámetros URL enviados a través de una petición GET estándar y devuelve el panorama de Street View. La figura 1.2 muestra un ejemplo de una respuesta de Google *Street View Image* API.

Una URL es utilizada siguiendo el siguiente formato:

```
https://maps.googleapis.com/maps/api/streetview?parameters
```

Tabla 2 Parámetros más importantes en Google Street View Image API

Parámetros más importantes en Google Street View Image API	
Parámetro	**Descripción**
location	Define la ubicación de la imagen más cercana tomada por el carro de Google Street View usando longitud y latitud o una dirección.
size	Define el tamaño de la imagen en pixeles (valor_horizontal x valor_vertical).
heading	Define la dirección de la brújula de la cámara usando valores de 0 a 360 grados.
pitch	Define el ángulo de inclinación de la cámara de Street View donde valores positivos inclinan la cámara hacia arriba mientras negativos hacia abajo.

Figura 2. Parámetros pitch y heading para la misma posición (latitud y longitud). (a) el parámetro pitch=25; (b) el parámetro pitch=0 y heading=253; (c) el parámetro heading=200

a) b) c)

1.2 Transferencia de Estado Representacional (REST)

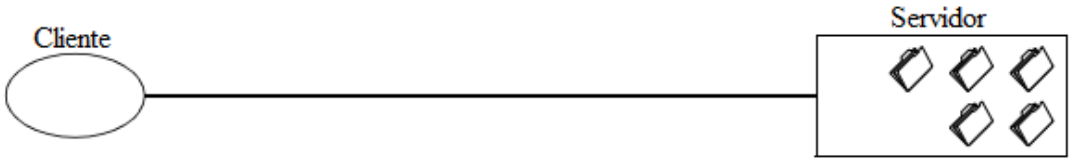
REST es un estilo híbrido derivado de estilos arquitectónicos basados en la red combinado con una serie de características adicionales que definen una interfaz de conector uniforme. Las características adicionales que componen REST se enuncian en las siguientes secciones y son mostradas en la figura 1.7 donde es posible observar todas las características y componentes de REST.

1.2.1 Característica Cliente-servidor de REST

El estilo arquitectónico cliente-servidor es frecuentemente el más encontrado en aplicaciones basadas en la red. Un servidor ofrece un conjunto de servicios y recibe las peticiones sobre esos servicios. Un cliente, con la necesidad de un servicio, envía una petición al servidor a través de un conector. El servidor rechaza o realiza la solicitud y envía una respuesta de vuelta al cliente. Una variedad de sistemas cliente-servidor han sido investigados por Sinha [15] y Umar [16].

La separación de los dominios es el principio detrás del estilo cliente-servidor. Una separación adecuada de funcionalidad simplifica el componente de servidor con el fin de mejorar la escalabilidad. Esta simplificación mueve toda la funcionalidad del usuario en el componente de cliente. La separación también permite que los dos tipos de componentes evolucionen de manera independiente. El estilo es mostrado en la figura 1.3.

Figura 3 Estilo Cliente-Servidor



Fuente: Architectural Styles and the Design of Network-based Software Architectures [11]

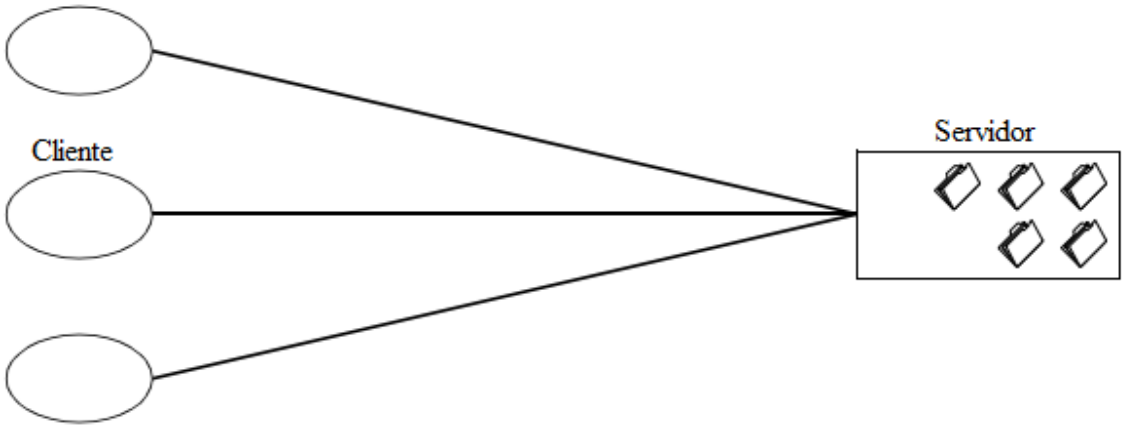
1.2.2 Característica Sin estado de REST

Un estilo Cliente-Servidor y sin estado no permite ningún estado de sesión en el servidor. Cada solicitud del cliente al servidor debe contener toda la información necesaria para comprender la solicitud y no se puede acceder a contextos almacenados en el servidor. El estado de la sesión se mantiene totalmente en el cliente.

Esta característica mejora las propiedades de visibilidad, fiabilidad y escalabilidad. La visibilidad se mejora debido a que un sistema de monitoreo no tiene que mirar más allá de una sola solicitud de

referencia con el fin de determinar la naturaleza completa de la solicitud. La fiabilidad se mejora, ya que facilita la tarea de recuperación de fallos parciales [17]. La escalabilidad se mejora porque al no tener que almacenar el estado entre solicitudes permite que el servidor atienda a los recursos libres rápidamente simplificando la implementación. La figura 1.4 muestra el estilo Cliente-sinEstado-Servidor.

Figura 4 Estilo Cliente-sinEstado-Servidor



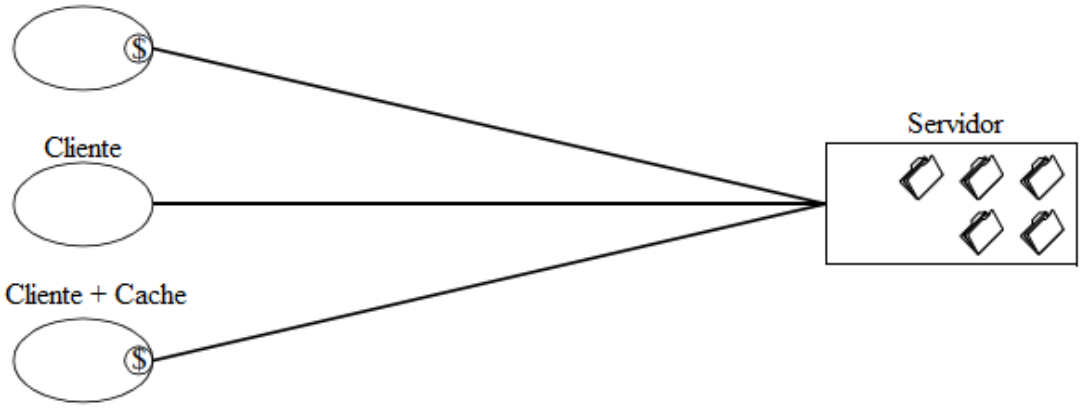
Fuente: Architectural Styles and the Design of Network-based Software Architectures [11]

1.2.3 Característica Caché de REST

Un caché actúa como un mediador entre el cliente y el servidor en el que las respuestas a solicitudes anteriores pueden volver a ser utilizadas en respuesta a solicitudes posteriores que son equivalentes y que puedan dar lugar a una respuesta idéntica a la de la memoria caché como si hubiera sido enviada por el servidor.

La ventaja de la adición de la memoria caché es que tienen el potencial de eliminar parcial o completamente algunas interacciones, mejorando la eficiencia y el rendimiento percibidos por el cliente. La figura 1.5 muestra el estilo Cliente-Caché-sinEstado-Servidor.

Figura 5 Estilo Cliente-Caché-sinEstado-Servidor

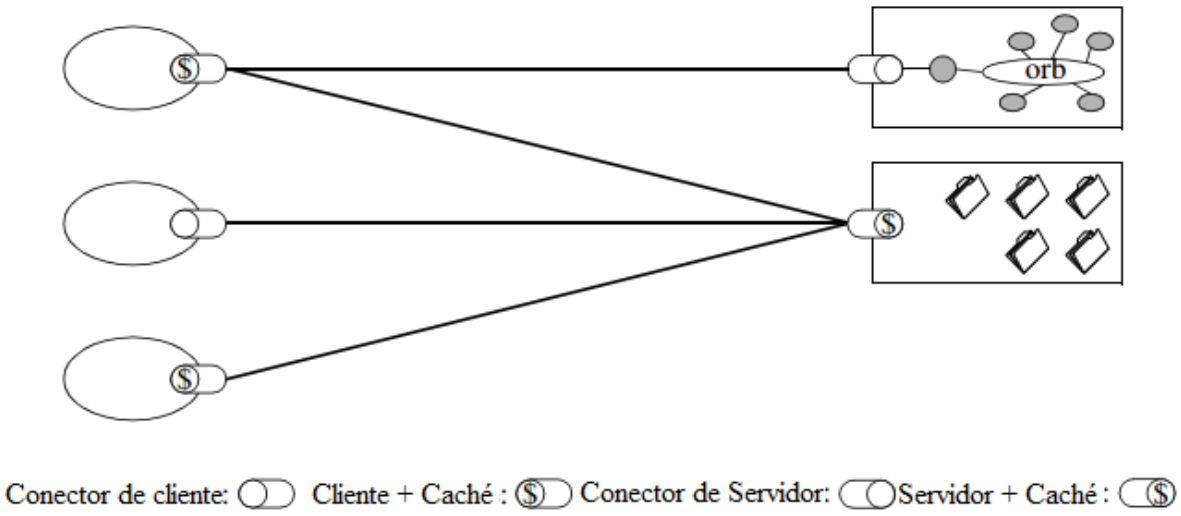


Fuente: Architectural Styles and the Design of Network-based Software Architectures [11]

1.2.4 Característica Interfaz uniforme de REST

La característica central que distingue a la arquitectura REST de otros estilos basados en la red es su enfoque en una interfaz uniforme entre los componentes. La arquitectura general del sistema se simplifica y la visibilidad de las interacciones se mejora. Las implementaciones son independientes de los servicios que prestan, lo que favorece la evolución independiente. La interfaz REST está diseñada para ser eficiente en la transferencia de datos hipermedia a gran tamaño, optimizando para el caso común de la Web. La figura 1.6 muestra el estilo Uniforme-Cliente-Caché-sinEstado-Servidor

Figura 6 Estilo Uniforme-Cliente-Caché-sinEstado-Servidor



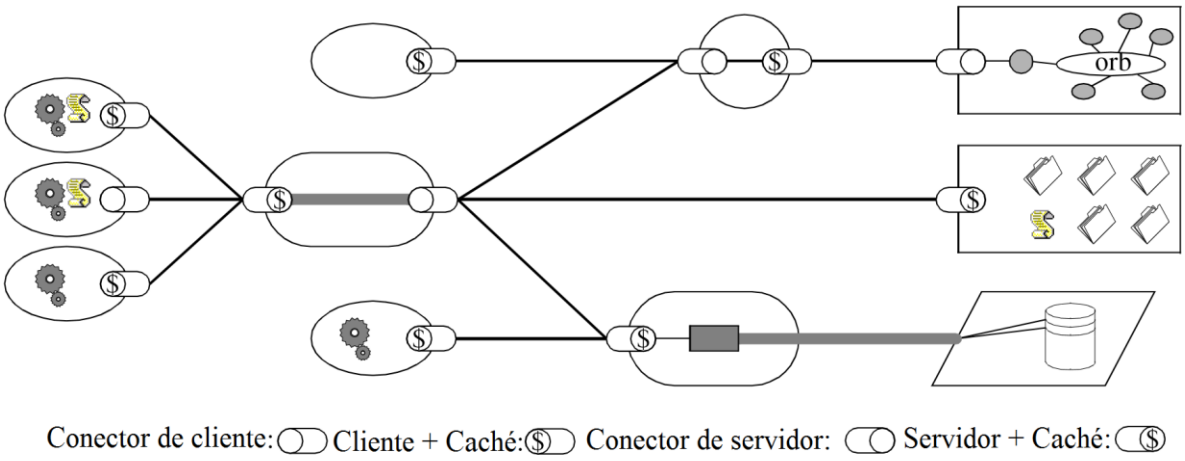
Fuente: Architectural Styles and the Design of Network-based Software Architectures [11]

1.2.5 Características Sistema de capas y Código bajo demanda de REST

Un sistema de capas está organizado jerárquicamente, cada capa presta servicios a la capa por encima de ella y usa servicios de la capa debajo de ella [18]. Su uso dentro de los sistemas basados en la red se limita a su combinación con el estilo cliente-servidor para proporcionar capas cliente-servidor.

En el estilo de código bajo demanda [19], un cliente tiene acceso a un conjunto de recursos, pero no los conocimientos sobre la forma de procesarlos. Se envía una petición a un servidor remoto para que envié el código que permite ejecutar los recursos, recibe ese código y lo ejecuta localmente.

Figura 7 Transferencia de Estado Representacional (REST)



Fuente: Architectural Styles and the Design of Network-based Software Architectures [11]

1.3 REST aplicado a HTTP

La extensión de este estilo de arquitectura es llamada RESTful. Los Sistemas RESTful normalmente, se comunican a través del protocolo de transferencia de hipertexto (HTTP) con los mismos verbos HTTP (GET, POST, PUT, DELETE, etc.) que los navegadores web utilizan para recuperar páginas web y enviar información a servidores remotos [20]. Las interfaces REST con sistemas externos utilizan recursos identificados por el identificador uniforme de recursos (URI), por ejemplo "/usuario/andres" que puede ser utilizada con el uso de verbos estándar como "GET /usuario/andres".

1.4 Geometría proyectiva

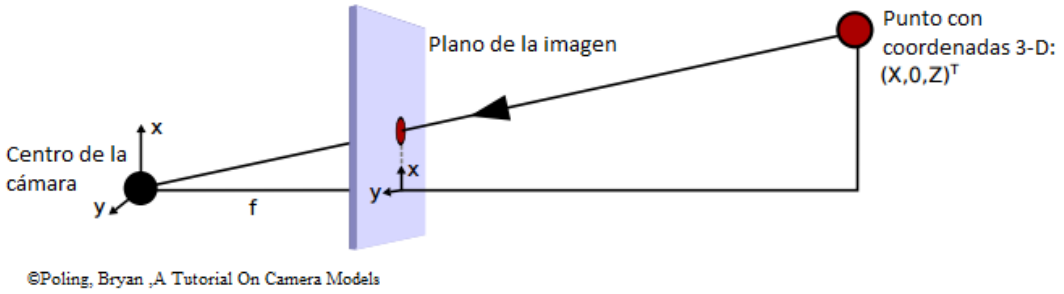
La geometría proyectiva es un marco matemático usado en visión por computador y en formación de imágenes [21]. En general, se dice que las imágenes son proyecciones bidimensionales de escenas tridimensionales. Un punto 2D (coordenadas de pixeles en una imagen) puede ser representado usando un par de valores $\boldsymbol{x} = (x, y) \in \mathbb{R}^2$. Los Puntos 2D también pueden ser representados usando coordenadas homogéneas $\boldsymbol{x} = (x, y, w) \in \mathcal{P}^2$ donde $\mathcal{P}^2 = \mathbb{R}^3 - (0,0,0)$ es llamado el espacio proyectivo 2D. Las letras mayúsculas $(\boldsymbol{X})$ representan puntos en el espacio y las letras minúsculas $(\boldsymbol{x})$ representan puntos en la imagen. El modelo de cámara estenopeica es el más utilizado en formación de imágenes ya que tiene en cuenta la proyectividad de la escena.

1.4.1 Modelo de cámara estenopeica

El modelo de la cámara estenopeica describe la relación matemática entre las coordenadas de un punto en 3D y su proyección sobre el plano de la imagen de una cámara estenopeica ideal, donde la apertura de la cámara se describe como un punto y ningún lente enfoca la luz [22]. En este modelo, se asume que el centro de la cámara está en el origen de un marco coordenado 3D. También, se asume que el plano de la imagen está posicionado paralelamente al plano xy, en la posición $z = f$ donde f es la distancia entre el centro óptico de la lente y el foco (distancia focal). Entonces, se define un sistema coordenado 2D en el plano de la imagen con el origen posicionado en $\boldsymbol{x} = (\,0\,,0\,,f\,)^T$ (en coordenadas euclidianas 3D).

Un punto en el espacio 3D $\boldsymbol{X} = (X, 0, Z)^T$ se proyecta en el plano de la imagen como se muestra en la figura 1.7 donde la formación de la imagen es modelada como una proyección de puntos en el espacio 3D a puntos 2D en la imagen.

Figura 8 proyección de un punto en el espacio 3D a un punto 2D en el plano de la imagen



Fuente: Architectural Styles and the Design of Network-based Software Architectures [22]

Usando triángulos semejantes y coordenadas homogéneas la proyección del punto de la figura 1.4 está dado por:

$$\boldsymbol{x} = P\boldsymbol{X}, \tag{1.1}$$

$$P = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \tag{1.2}$$

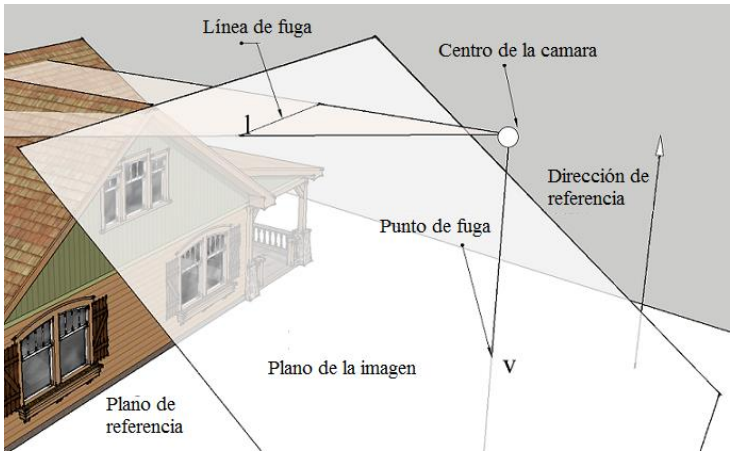
donde P es llamada “Matriz de proyección”.

1.5 Metrología de una vista

Usando metrología de una vista es posible estimar medidas 3D en una imagen de una escena teniendo sólo mínima información geométrica determinada de la imagen [8]. Esta información mínima es usualmente la línea de fuga de un plano de referencia (ver figura 1.10), y un punto de fuga para una dirección perpendicular a este plano, es decir, no es necesario conocer los parámetros internos de la cámara (por ejemplo, longitud de foco) ni de la explicita relación entre la cámara y el mundo (pose).

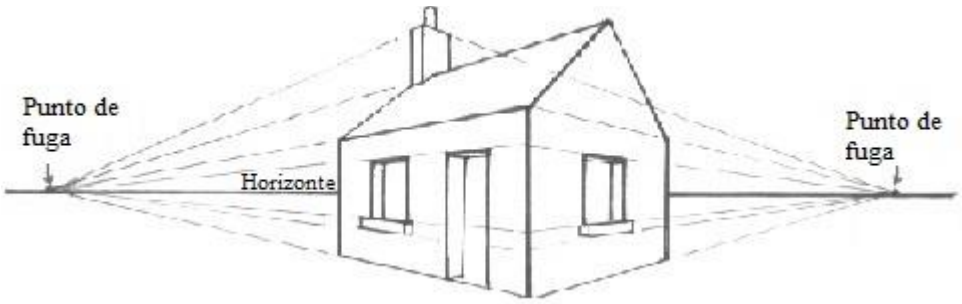
El modelo de cámara empleado es el modelo de cámara estenopeica. Este modelo asume que la línea de fuga de un plano de referencia en la escena puede ser calculada de la imagen junto con un punto de fuga para otra dirección no paralela al plano de referencia. La geometría básica se muestra en la figura 1.9.

Figura 9: Geometría básica: La línea de fuga **l** del plano de referencia que en este caso es el suelo, es la intersección del plano de la imagen con un plano paralelo al suelo en la dirección de referencia y pasando a través del centro de la cámara. El punto de fuga **v** es la intersección del plano de la imagen con una línea paralela a la dirección de referencia a través del centro de la cámara.



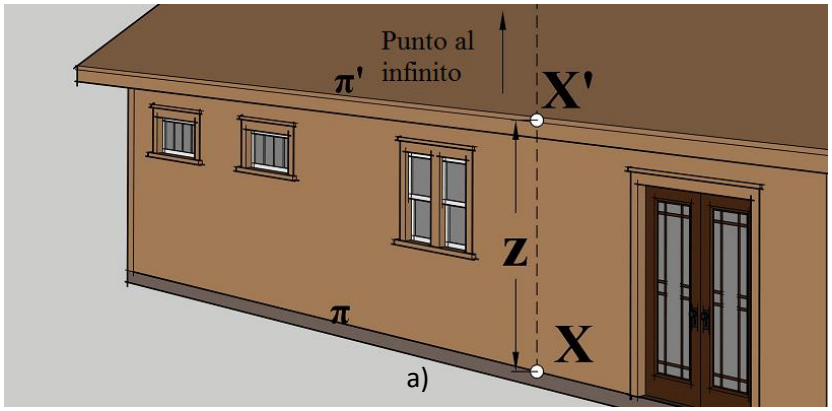
Fuente: Adaptada de Single View Metrology [8]

Figura 10 Horizonte o línea de fuga y puntos de fuga

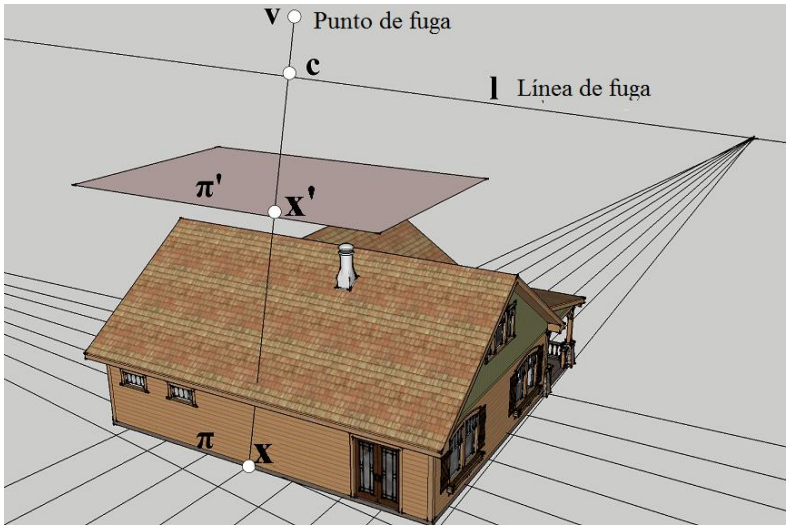


La distancia a estimar entre los puntos $\boldsymbol{x}$ y $\boldsymbol{x}'$ es mostrada en la figura 1.7 (b). Dada la línea de fuga del plano del suelo y el punto de fuga para una dirección vertical, la distancia entre un punto del suelo y cualquier otro punto vertical a este puede ser estimada con un factor de escala. El factor de escala puede ser determinado con la distancia de la cámara al suelo.

Figura 11 Distancia entre dos planos relativa la distancia del centro de la cámara a uno de los planos: (a) en el mundo; (b) en la imagen. El punto $\boldsymbol{x}$ en el plano $\boldsymbol{\pi}$ corresponde al punto $\boldsymbol{x}'$ en el plano $\boldsymbol{\pi}'$. Los cuatro puntos alineados $\boldsymbol{v}, \boldsymbol{x}, \boldsymbol{x}'$y la intersección $\boldsymbol{c}$ de la línea uniéndolos con la línea de fuga define la razón doble (*Cross ratio* en inglés). El valor de la razón doble determina la relación de distancias entre planos en el mundo.



a)



b)

Fuente: Adaptada de Single View Metrology [8]

Los cuatro puntos $\boldsymbol{x}, \boldsymbol{x}', \boldsymbol{c}, \boldsymbol{v}$ mostrados en la figura 1.7 (b) definen una razón doble. El punto de fuga es la imagen de un punto al infinito en la escena y el punto $\boldsymbol{c}$ es la imagen de un punto a la distancia Z_c del plano π, donde Z_c es la distancia del centro de la cámara al plano π. En el mundo real el valor de la razón doble define una relación de longitud afín la cual determina la distancia Z entre los planos conteniendo $\boldsymbol{X}'$ y $\boldsymbol{X}$ relativa a la distancia de la cámara Z_c del plano π.

Capítulo 2

MÉTODOS TRADICIONALES DE ESTIMACIÓN DE ALTURAS EN EDIFICIOS.

2.1 Métodos basados en la sombra de los edificios

El uso de sombras para estimar la altura de objetos en imágenes aéreas es una técnica clásica en fotogrametría y en técnicas manuales de interpretación de fotos [23]. Las hipótesis principales indican que los objetos a medir son verticales, que las sombras se proyectan desde la parte superior del objetivo y no a los lados, y que las sombras caen en terreno plano. El uso de las medidas de las sombras y su inversa, la determinación del ángulo de elevación del sol usando objetos de altura conocida son técnicas ampliamente usadas por interpretadores de fotos.

El primer paso de este método es la extracción de la regiones de sombra, el cual se realiza usando un conjunto simple de técnicas de procesamiento de imágenes que incluyen suavizado, valor umbral y extracción de regiones conectadas. Luego, se estima la dirección del sol con base en un análisis estadístico de sus límites de sombra/estructura o de una base de datos [24] .Además, la elevación angular puede ser determinada basándose en el conocimiento de la hora del día en el cual la imagen fue tomada, su latitud y la declinación del sol corregida a la hora promedio de Greenwich [24]. Por último, la altura de los edificios se estima aplicando una relación trigonométrica entre el ángulo de inclinación del sol y la longitud de su sombra dada por

$$altura(L, \phi) = L \tan(\varphi), \tag{2.1}$$

donde L es la longitud de la sombra y ϕ es el ángulo de inclinación del sol.

2.2 Métodos basados en telémetros y sensores LiDAR

En adición a las técnicas de fotogrametría que confían en imágenes aéreas, la generación de modelos 3D a partir de nubes de puntos proporcionadas por sensores LIDAR (*Ligth detection and ranging* por su sigla en inglés) y telémetros es de gran importancia en la comunidad científica de la teledetección. Este desarrollo se debe en mayor parte al progreso en las tecnologías de sensores las cuales han hecho posible la adquisición de nubes de puntos muy densas usando escáneres laser transportados en vehículos aéreos. Usando datos LIDAR con densidades de punto de hasta un punto por metro cuadrado, es posible no solo detectar edificios y sus alturas sino también crear modelos 3D. Esta tecnología permite determinar la distancia desde un emisor láser a un edificio utilizando un haz láser pulsado. La distancia al objeto se determina midiendo el tiempo de retraso entre la emisión del pulso y su detección a través de la señal reflejada [25].

2.3 Métodos basados en imágenes SAR (Synthetic aperture radar)

Los Sensores SAR pueden adquirir imágenes independientes de condiciones meteorológicas y de las condiciones de luz. Los Nuevos sensores de alta resolución espacial (VHR) SAR a bordo de los satélites TerraSAR-X y COSMO-SkyMed que fueron lanzados en 2007, proporcionan imágenes SAR con resolución espacial de hasta 1 m. Varias técnicas en la estimación de alturas han sido ya propuestas para imágenes VHR SAR en la literatura. En [26-28] se proponen métodos semiautomáticos para la estimación de altura en imágenes VHR SAR por medio de la sombra o el análisis de escala, mientras que los métodos en (29-31) hacen uso de SAR interferómetrico (InSAR). El uso de SAR estereoscópico (radargrametría) ha sido propuesto en [32] y [33]. Recientemente en [34] y [35] se propusieron los métodos basados en datos SAR multiaspecto, en los que la misma área es vista desde diferentes rutas de vuelo.

Capítulo 3

MÉTODO PROPUESTO USANDO IMÁGENES DE STREET-VIEW Y METROLOGÍA DE UNA VISTA

El modelo propuesto busca aprovechar el conjunto de imágenes de Google Street-View para estimar la altura de los edificios de cualquier ciudad usando metrología de una vista reduciendo los altos costos de los métodos tradicionales, además un algoritmo de visualización muestra los resultados en un mapa de Google. Por esto, se desarrolló un sistema REST que permitiera utilizar las funciones API disponibles para acceder a todo el conjunto de imágenes con los parámetros adecuados para permitir la estimación de la altura de los edificios. La figura 3.1 presenta el diagrama de flujo de todo el algoritmo descrito anteriormente.

3.1 Sistema de transferencia de estado representacional (REST)

Un sistema de transferencia de estado representacional (REST) es definido para obtener el mapa del área de la ciudad donde se realizará la estimación de las alturas de los edificios, para establecer la ruta y coordenada GPS correcta de las imágenes de Google Street-View y obtener sus imágenes correctamente.

El sistema REST es definido como sigue:

$$Map = SMA(center, maptype, style) \tag{3.1}$$

$$\overrightarrow{v_2} = RA(\overrightarrow{v_1}) \quad \overrightarrow{v_1} \in \mathcal{R}^{100x3} \tag{3.2}$$

$$StreetImage = SIA(location, heading, pitch) \tag{3.3}$$

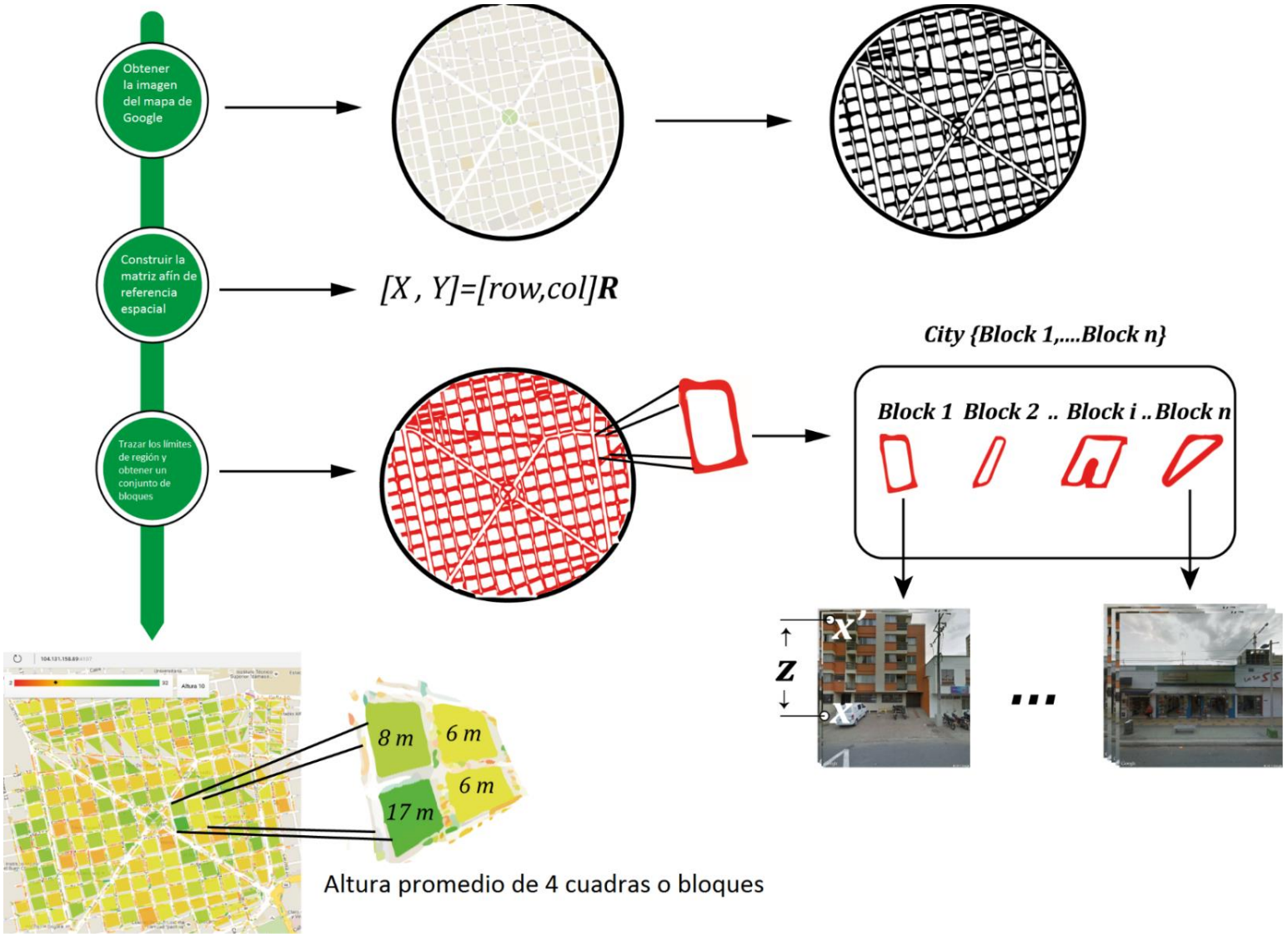
Donde SMA (Static Map API), RA (Roads API) y SIA (Street Image API) son los métodos de petición basados en parámetros URL definidos en la sección 1.1.

Usando (3.1) y los puntos GPS de sus esquinas la matriz afín de referencia espacial puede ser construida como

$$[lon\ lat] = [row\ col\ 1]\ \boldsymbol{R}, \tag{3.4}$$

donde $\boldsymbol{R}$ es una matriz afín de transformación 3x2 que mapea índices de filas y columnas de una imagen o datos de una cuadrícula regular a coordenadas de mapa 2D o coordenadas geográficas (longitud y latitud geodésica). Usando esta matriz es posible transformar pixeles (fila, columna) en coordenadas geográficas (longitud, latitud).

Figura 12 Diagrama de flujo del algoritmo propuesto



Las técnicas de procesamiento de imágenes y el algoritmo de traza de Moore-Neighbor con el criterio de parada de Jacob's pueden ser usados para trazar los límites de región de la imagen del mapa, resultando en un conjunto de bloques o cuadras de la ciudad dados por

$$City = \{Block_1, Block_2, .. Block_i \dots Block_m\},$$
$$donde\ Block_i = \{I_{i1}, I_{i2}, .. I_{ij} .. I_{in}\},$$

donde I_{ij} es la imagen de Google Street-View del bloque o cuadra i obtenida usando (3.3). $Block_i$ es obtenido usando (3.2). Para cada imagen la altura promedio es estimada usando metrología de una vista discutida en la sección 3.3. Entonces la matriz de alturas de una ciudad puede ser definida como

$$A = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{ij} & \cdots & h_{1n} \\ h_{21} & h_{22} & \cdots & h_{2j} & \cdots & h_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ h_{i1} & h_{i2} & \cdots & h_{ij} & \cdots & h_{in} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{m1} & h_{m2} & \cdots & h_{mj} & \cdots & h_{mn} \end{bmatrix}, \tag{3.5}$$

donde h_{ij} es la altura de la imagen j del bloque o cuadra i.

3.2 GeoJSON

GeoJSON [36] es un formato estándar abierto para para la codificación de características geográficas, basado en *JavaScript Object Notation* [37]. Un objeto GeoJSON representa tipos geométricos como puntos, polígonos o una colección de ellos. En este trabajo, GeoJSON es usado para representar la altura de los edificios y la altura promedio de los bloques o cuadras.

Los mapas de Google proporcionan un contenedor de capas para cualquier tipo de dato geoespacial, esto permite mostrar datos GeoJSON. La función *GeoJsonBlock* crea un GeoJSON del $Block_i$ con el promedio de las alturas de los edificios que lo componen para derivar el paso final en el diagrama de flujo de la figura 3.1.

3.3 Metrología de una vista

La altura de un edificio puede ser estimada basándose en puntos de fuga y la altura de la cámara [8]. Un sistema coordenado afín XYZ es definido en el espacio donde el origen está en el suelo con los ejes X y Y abarcando el plano y el eje Z es la dirección de referencia. El sistema coordenado afín de la imagen es xy y un punto $\boldsymbol{X}$ en el espacio es proyectado a un punto $\boldsymbol{x}$ en la imagen a través de la matriz de proyección $P \in \mathbb{R}^{3x4}$ como

$$\boldsymbol{x} = \boldsymbol{PX} = [\boldsymbol{P_1} \quad \boldsymbol{P_2} \quad \boldsymbol{P_3} \quad \boldsymbol{P_4}]\boldsymbol{X}, \tag{3.6}$$

donde $\boldsymbol{x}$ y $\boldsymbol{X}$ son vectores homogéneos y "=" significa igualdad en escala.

Si los puntos de fuga son denotados $\boldsymbol{v}_X, \boldsymbol{v}_y$ y $\boldsymbol{v}$ para las direcciones X, Y y Z respectivamente, entonces las tres primeras columnas de P son los puntos de fuga $\boldsymbol{v}_X = \boldsymbol{P}_1$, $\boldsymbol{v}_y = \boldsymbol{P}_2$, $\boldsymbol{v} = \boldsymbol{P}_3$ y la columna final de P es la proyección del origen del sistema coordenado del espacio $\boldsymbol{o} = \boldsymbol{P}_4$. Ya que la elección de sistema coordenado tiene los ejes X y Y en el plano de referencia $\boldsymbol{P_1} = \boldsymbol{v}_X$ y $\boldsymbol{P}_2 = \boldsymbol{v}_y$ son dos puntos distintos de la línea de fuga. La línea de fuga es denotada por $\boldsymbol{I}$ y se especifica que los puntos de fuga $\boldsymbol{v}_X$ y $\boldsymbol{v}_y$ pasan por esta línea, denotados ahora como ${\boldsymbol{I}_1}^T$ y ${\boldsymbol{I}_2}^T$, con ${\boldsymbol{I}_i}^T = 0$.

La columna final (el origen del sistema coordenado del espacio) no debe estar en la línea de fuga, ya que si lo hiciera entonces las tres columnas serían puntos en la línea de fuga y entonces no serían linealmente independientes, por lo tanto $\boldsymbol{P}_4 = \frac{\boldsymbol{I}}{\| \boldsymbol{I} \|} = \overline{\boldsymbol{I}}$.

Por lo tanto, la parametrización final de la matriz de proyección P es

$$\boldsymbol{P} = [{\boldsymbol{I}_1}^T \ {\boldsymbol{I}_2}^T \quad \alpha\boldsymbol{v} \ \ \overline{\boldsymbol{I}}], \tag{3.7}$$

donde α es un factor de escala. Ahora, si se asume que el centro de la cámara (ver figura 1.9) está dado por $\boldsymbol{C} = (X_{\boldsymbol{c}}, \ Y_c, \ Z_c, \ W_c)$, entonces $P\boldsymbol{C} = 0$

$$P\boldsymbol{C} = \boldsymbol{P_1}X_c + \boldsymbol{P}_2Y_c + \boldsymbol{P_3}Z_c + \boldsymbol{P}_4W_{\boldsymbol{c}} = 0 \tag{3.8}$$

La solución a este conjunto de ecuaciones con α desconocido es dado (usando regla de Cramer) por

$$\begin{aligned} X_c &= -\det[{\boldsymbol{I}_2}^T \quad \boldsymbol{v} \quad \overline{\boldsymbol{I}}] \\ Y_c &= \det[{\boldsymbol{I}_1}^T \quad \boldsymbol{v} \quad \overline{\boldsymbol{I}}] \\ \alpha Z_c &= -\det[{\boldsymbol{I}_1}^T \quad {\boldsymbol{I}_2}^T \quad \overline{\boldsymbol{I}}] \\ W_c &= \det[{\boldsymbol{I}_1}^T \quad {\boldsymbol{I}_2}^T \quad \boldsymbol{v}] \end{aligned} \tag{3.9}$$

Usando $Z_c = 2.5m$ que es la altura aproximada de la cámara en el carro de Google de acuerdo a Google Earth es posible hallar α.

Dado el plano de referencia del edificio en el suelo, su altura $\boldsymbol{Z}$ puede ser entonces encontrada por la siguiente ecuación

$$\boldsymbol{Z} = -\frac{\|\operatorname{cross}(\boldsymbol{x}, \boldsymbol{x}')\|}{(\boldsymbol{I} \cdot \boldsymbol{x})\|\operatorname{cross}(\alpha \boldsymbol{v}, \boldsymbol{x}')\|} \tag{3.10}$$

donde $\boldsymbol{x}$ y $\boldsymbol{x}'$ son la parte baja y la parte alta del edificio respectivamente (ver figura 1.11) y Cross(x,y) es el producto cruz de x con y.

Prueba de la ecuación (3.10) puede ser encontrada en [8].

La matriz P es calculada manualmente para todas las imágenes de Street-View con $pitch = 0$, así mismo otra matriz P para imágenes con $pitch > 25$.

Algoritmo 1 ($\boldsymbol{Altura}$): Estimación de la altura promedio usando metrología de una vista

---

Entrada: Imagen de Google Street-View(GSV) con el parámetro $heading$ correcto y su coordenada GPS.

Salida: Altura promedio de los edificios en la imagen.

Detección de la región del suelo.

$W =$ Tamaño (GSV)

$pitch = 0$

Para $i = 1$ **hasta** $\frac{w}{80}$ **hacer**

Mientras el cielo no pueda ser detectado **hacer**

$pitch = pitch + 5$

GSV= $SIA(GPS, heading, pitch)$

Fin mientras

$Z(i) = -\frac{\|\operatorname{cross}(\boldsymbol{x}, \boldsymbol{x}')\|}{(\boldsymbol{I} \cdot \boldsymbol{x})\|\operatorname{cross}(\alpha \boldsymbol{v}, \boldsymbol{x}')\|}$

Fin para

$Salida = promedio(Z)$

Algoritmo 2 Estimación de la altura de los edificios de una ciudad

Entrada: Punto GPS del centro de la ciudad y los cuatro puntos GPS de sus esquinas.

Salida: Matriz de alturas A y la colección GeoJSON.

$Mapa = SMA(center, terrain, off)$

Construir la matriz afín de referencia espacial $\boldsymbol{R}$

$City = TrazarLimites(Mapa)$

$m = tamaño(City)$

Para $i = 1$ **hasta** m **hacer**

$BloqueImagen = City[i]$

$n = tamaño(\ BloqueImagen)$

Para $j = 1$ **hasta** $\frac{n}{100}$ **hacer**

$\overrightarrow{v_1} = \ BloqueImagen[(j * 100 + 1 : (j + 1) * 100)]$

$\overrightarrow{v_2} = RA(\ [\overrightarrow{v_1} \quad 1]\boldsymbol{R}\ )$

Para $k = 2$ **hasta** 100 **hacer**

$heading =$ Azimuth ($\overrightarrow{v_2}(k-1), \overrightarrow{v_2}(k)$) + 90

GSV= $SIA(\ \overrightarrow{v_2}(k-1), heading\ , 0\ )$

$\boldsymbol{A}(i, k-1) = Altura(\ \text{GSV}, heading, \overrightarrow{v_2}(k-1)\ )$

$MitadAnterior = \frac{\overrightarrow{v_2}(k-1)+\overrightarrow{v_2}(k-2)}{2}$

$MitadSiguiente = \frac{\overrightarrow{v_2}(k-1)+\overrightarrow{v_2}(k)}{2}$

Agregar polígono a la colección con las mitades y asignar la altura $\boldsymbol{A(i, k-1)}$

Fin para

Fin para

GeoJsonBlock($Block_i, \boldsymbol{A(i, :)}$)

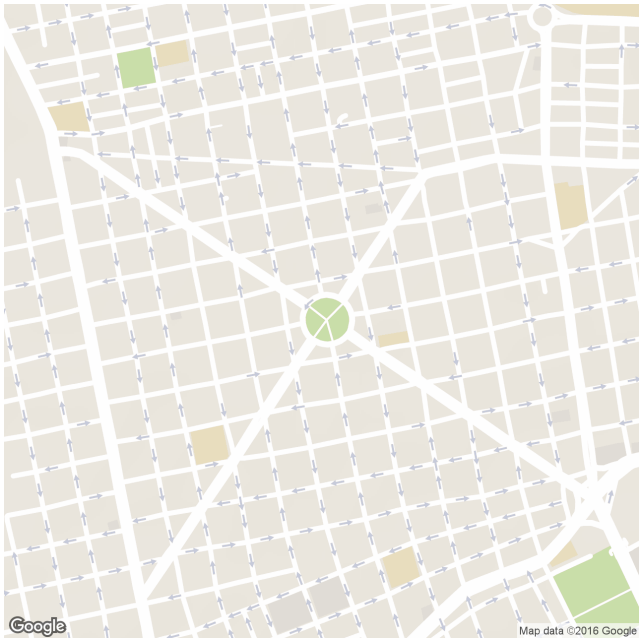
Fin para

Capítulo 4

Resultados

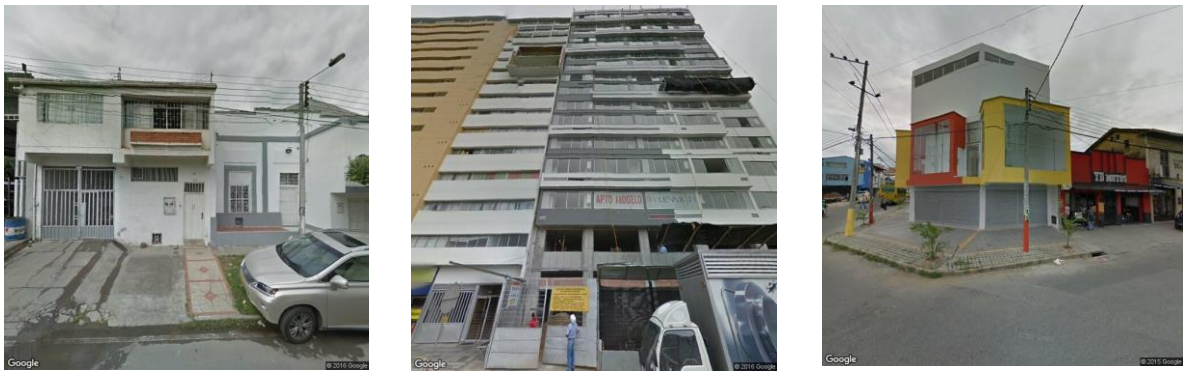
El desempeño del modelo propuesto es evaluado realizando un muestreo aleatorio de alturas estimadas por el algoritmo para luego compararlas con las de las alturas reales conocidas de los edificios. La ciudad escogida es Bucaramanga en la zona norte y un área de la ciudad de Madrid. El área donde se ejecutará el algoritmo es ilustrada en la figura 4.1.

Figura 13 Área de la ciudad de Bucaramanga



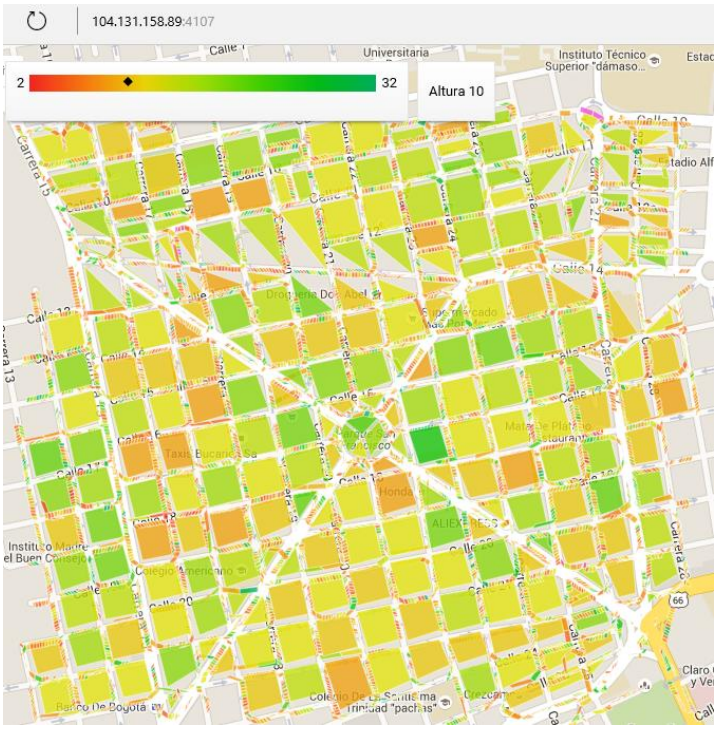
El sistema REST propuesto, descrito en el Algoritmo 2, trabaja sobre toda la imagen extrayendo cada bloque o cuadra de la ciudad creando el conjunto $city$. Luego, para obtener las coordenadas GPS de toda el área es usada la función (3.2) y la matriz $\boldsymbol{R}$ a cada cuadra de la ciudad. Las coordenadas son argumento de la función (3.3) para obtener las imágenes de Street-View. Finalmente, calculando el azimut entre puntos y sumándole 90 grados se encuentra el $heading$ adecuado y con esto usar la metrología de una vista para estimar la altura promedio de los edificios en la imagen. En la figura 4.2 se observan 3 estimaciones de edificios con diferentes alturas. La colección de GeoJSON y su visualización en un mapa de Google es mostrada en la figura 4.3.

Figura 14 Resultado de la estimación para el área norte de Bucaramanga. (a) la altura estimada es 4.5 metros. (b) la altura estimada es 30.07 metros. (c) la altura estimada es 10.99 metros.



a) b) c)

Figura 15 Visualización de la capa de alturas sobre Google Maps creada por el algoritmo propuesto. El mapa puede ser accedido desde internet.



Cuantitativamente, se realizó un muestreo aleatorio sobre todas la imágenes de Google Street-View donde se compara la altura real del edificio estimada manualmente y la obtenida por el algoritmo. Los resultados se muestran en la tabla 4.1. Además, de la muestra aleatoria de 73 edificios se obtuvo un error promedio de 1.38 metros.

Tabla 3: Resultados experimentales para el área de Bucaramanga. En la tabla se muestran la altura estimada por el algoritmo, la altura estimada manualmente usando como referencia la medida de cada piso de 2.7 metros y la diferencia entre las dos medidas

Altura del piso 2.7 m		
Estimada	**Estimada manualmente**	**Diferencia**
3.0609	3.024	0.0369
4.1584	4.239	0.0806
5.2569	5.4	0.1431
8.2441	8.1	0.1441
10.9957	10.8	0.1957
2.4794	2.7	0.2206
6.1658	5.94	0.2258
5.6305	5.4	0.2305
7.8053	7.56	0.2453
9.3235	9	0.3235
5.2944	4.86	0.4344
2.8857	3.321	0.4353
3.9566	3.51	0.4466
4.5637	4.086	0.4777
8.5795	8.1	0.4795
6.1546	5.67	0.486
5.4492	5.94	0.4908
4.8622	5.4	0.5378
4.33	3.757	0.5725

Figura 16 Área de la ciudad de Madrid

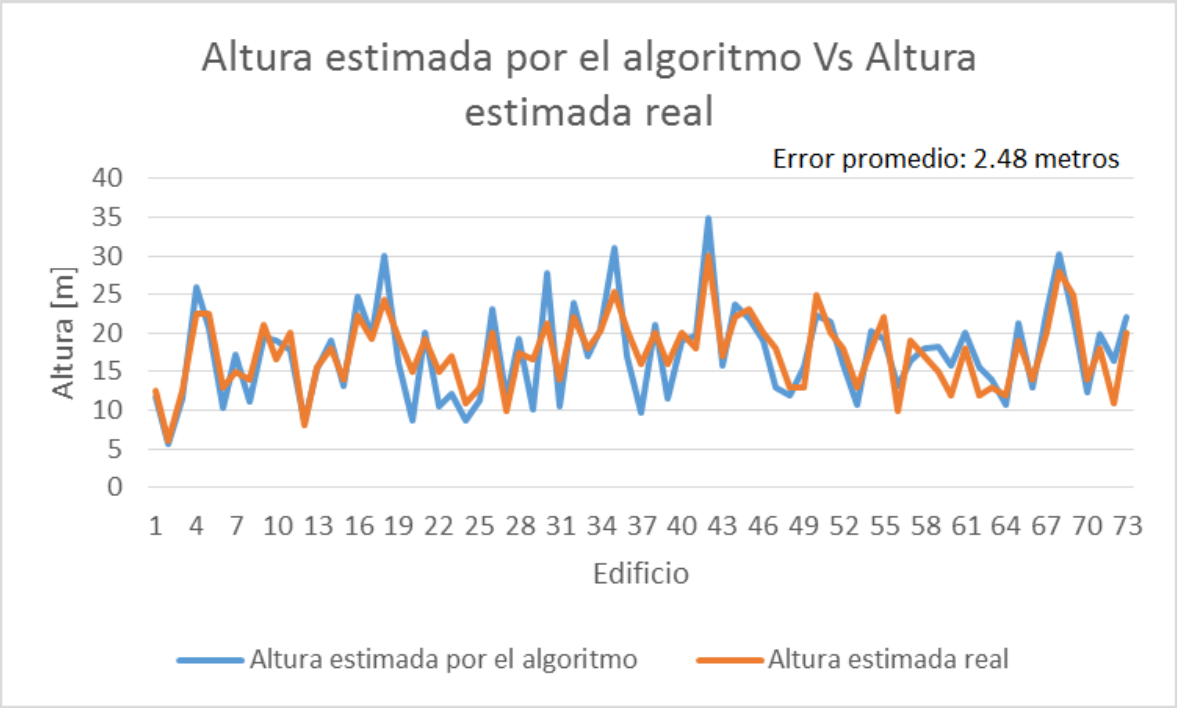
Figura 17 Resultado de la estimación para el área de la ciudad de Madrid. (a) la altura estimada es 19 metros. (b) la altura estimada es 23 metros. (c) la altura estimada es 11.6 metros.

a) b) c)

Figura 18 Visualización de la capa de alturas sobre Google Maps creada por el algoritmo propuesto. El mapa puede ser accedido desde internet.

Cuantitativamente, se realizó un muestreo aleatorio sobre todas la imágenes de Google Street-View donde se compara la altura real del edificio estimada manualmente y la obtenida por el algoritmo. Los resultados se muestran en la figura 4.7.

Figura 19 Altura estimada por el algoritmo vs Altura estimada real.



Capítulo 5

Conclusiones

5.1 Conclusiones

- Se desarrolló un algoritmo que obtiene ordenadamente a través de REST las cuadras dada un área de cualquier ciudad para acceder a las imágenes de Google Street-View y obtener su altura promedio usando metrología de una vista.

- El algoritmo propuesto ahorra el costo elevado de los métodos tradicionales porque al usar imágenes Google *Street-View* no es necesario utilizar ningún hardware especializado.

- El algoritmo propuesto detecta las alturas del área norte de Bucaramanga con un error promedio de 1.38 metros y un área de la ciudad de Madrid con un error promedio de 2.48 metros.

- Se descubrió que la principal fuente de error del algoritmo recae en la incorrecta detección del piso y el cielo de la imagen donde el edificio va a ser estimado. Además, entre más alto es el edificio los errores de perspectiva afectan la precisión del algoritmo.

- Se propuso un algoritmo que extrae las imágenes de Google *Street-View* para su uso en general. En este trabajo, se usaron para estimar la altura de los edificios, sin embargo, pueden ser utilizadas para diversas aplicaciones desde el campo de la visión por computador hasta áreas donde es importante investigar y analizar el ambiente de las ciudades.

Referencias

[1] Rundle, A. G., Bader, M. D. M., Richards, C. A., Neckerman, K. M., & Teitler, J. O. (2011). Using Google Street View to Audit Neighborhood Environments. AMEPRE, 40(1), 94–100. http://doi.org/10.1016/j.amepre.2010.09.034.

[2] Kelly, C. M., Wilson, J. S., Baker, E. A., Miller, D. K., & Schootman, M. (2013). Using Google Street View to Audit the Built Environment : Inter-rater Reliability Results, 45, 108–112. http://doi.org/10.1007/s12160-012-9419-9

[3] Flores, A., & Belongie, S. (2010). Removing pedestrians from Google Street View images, 53–58.

[4] Tsai, V. J. D., & Chang, C. (2013). Three-dimensional positioning from Google street view panoramas, (November 2012), 229–239. http://doi.org/10.1049/iet-ipr.2012.0323.

[5] Tom, M. H. (2009). From Google Street View to 3D City Models, 2188–2195.

[6] Yan, W. Y. (2013). Potential Accuracy of Traffic Signs ' Positions Extracted From Google Street View, 14(2), 1011–1016.

[7] Xiao, J., Bay, C. W., & Kong, H. (2009). Multiple View Semantic Segmentation for Street View Images, (Iccv).

[8] Criminisi, A., Reid, I., & Zisserman, A. (2000). Single view metrology. International Journal of Computer Vision, 40(2), 123–148. Retrieved from http://link.springer.com/article/10.1023/A:1026598000963.

[9] Google., «Google Street View,» [En línea]. Available: https://www.google.com/maps/streetview/. [Último acceso: 2 12 2015].

[10] Google, «APIs Explorer,» [En línea]. Available: https://developers.google.com/apis-explorer/#p/. [Último acceso: 8 12 2015].

[11] Google, «Google Static Maps Developer Guide,» [En línea]. Available: https://developers.google.com/maps/documentation/static-maps/intro. [Último acceso: 2 12 2015].

[12] http://maps.googleapis.com/maps/api/staticmap?center=7.131183,-73.125125&zoom=16&scale=2&size=800x800&maptype=terrain&style=feature:all|element:labels|visibility:off&key=AIzaSyCND3owf_Nq90qHZrNvSdq-f9H25gO0x9k.

[13] Google, «Introduction to the Google Maps Roads API,» [En línea]. Available: https://developers.google.com/maps/documentation/roads/intro. [Último acceso: 2 12 2015].

[14] Google, «Google Street View Image API,» [En línea]. Available: https://developers.google.com/maps/documentation/streetview/?hl=en. [Último acceso: 7 12 2015].

[15] A. Sinha. Client-server computing. Communications of the ACM, 35(7), July 1992, pp. 77-98)] y Umar [16 (A. Umar. Object-Oriented Client/Server Internet Environments. Prentice Hall PTR, 1997.

[16] Fielding, R. T. (2000). Architectural Styles and the Design of Network-based Software Architectures. CALIFORNIA

[17] J. Waldo, G. Wyant, A. Wollrath, and S. Kendall. A note on distributed computing. Technical Report SMLI TR-94-29, Sun Microsystems Laboratories, Inc., Nov. 1994

[18] D. Garlan and M. Shaw. An introduction to software architecture. Ambriola & Tortola (eds.), Advances in Software Engineering & Knowledge Engineering, vol. II, World Scientific Pub Co., Singapore, 1993, pp. 1-39.

[19] A. Fuggetta, G. P. Picco, and G. Vigna. Understanding code mobility. IEEE Transactions on Software Engineering, 24(5), May 1998, pp. 342-361.

[20] Fielding, Roy Thomas (2000). "Chapter 5: Representational State Transfer (REST)". Architectural Styles and the Design of Network-based Software Architectures (Ph.D.). University of California, Irvine. This chapter introduced the Representational State Transfer (REST) architectural style for distributed hypermedia systems. REST provides a set of architectural constraints that, when applied as a whole, emphasizes scalability of component interactions, generality of interfaces, independent deployment of components, and intermediary components to reduce interaction latency, enforce security, and encapsulate legacy systems.

[21] A. Z. Richard Hartley, Multiple View Geometry in Computer Vision, New York, 2004.

[22] B. Poling, A Tutorial On Camera Models.

[23] T. E. Awry, interpretation of Aerial Photographs Minneapolis. MN: Burgess Publishing Company, 1977

[24] D. M. McKeoun. “MAPS: The organization of a spatial database system using imagery. Terrain, and map data,” in Proc. DARPA Image Understanding Workshop, June 1983. pp. 105-127

[25] A NEW METHOD FOR BUILDING EXTRACTION IN URBAN AREAS FROM HIGH-RESOLUTION LIDAR DATA.

[26] M. Eineder, H. Breit, T. Fritz, B. Schättler, and A. Roth, "TerraSAR-X SAR products and processing algorithms," in Proc. IEEE IGARSS, Seoul, Korea, 2005, vol. 7, pp. 4870–4873

[27] A. J. Bennett and D. Blacknell, "Infrastructure analysis from high resolution SAR and InSAR imagery," in Proc. 2nd GRSS/ISPRS Joint Workshop Remote Sens. Data Fusion Over Urban Areas, Berlin, Germany, May 22–23, 2003, pp. 230–235

[28] F. Tupin, "Extraction of 3D information using overlay detection on SAR images," in Proc. 2nd GRSS/ISPRS Joint Workshop Remote Sens. Data Fusion Over Urban Areas, Berlin, Germany, May 22–23, 2003, pp. 72–76

[29] P. Gamba, B. Houshmand, and M. Saccani, "Detection and extraction of buildings from interferometric SAR data," IEEE Trans. Geosci. Remote Sens., vol. 38, no. 1, pp. 611–618, Jul. 2000

[30] P. Gamba and B. Houshmand, "Digital surface models and building extraction: A comparison of IFSAR and LIDAR data," IEEE Trans. Geosci. Remote Sens., vol. 38, no. 4, pp. 1959–1968, Jul. 2000

[31] C. Tison, F. Tupin, and H. Maître, "A fusion scheme for joint retrieval of urban height map classification from high-resolution interferometric SAR images," IEEE Trans. Geosci. Remote Sens., vol. 45, no. 2, pp. 496–505, Feb. 2007

[32] E. Simonetto, H. Oriot, and R. Garello, "Rectangular building extraction from stereoscopic airborne radar images," IEEE Trans. Geosci. Remote Sens., vol. 43, no. 10, pp. 2386–2395, Oct. 2005.

[33] R. Hill, C. Moate, and D. Blacknell, "Estimating building dimensions from synthetic aperture radar image sequences," IET Radar, Sonar Navig., vol. 2, no. 3, pp. 189–199, Jun. 2008

[34] F. Xu and Y.-Q. Jin, "Automatic reconstruction of building objects from multiaspect meter-resolution SAR images," IEEE Trans. Geosci. Remote Sens., vol. 45, no. 7, pp. 2336–2353, Jul. 2007

[35] S. Sauer, L. Ferro-Famil, A. Reigber, and E. Pottier, "Multi-aspect POLInSAR 3D urban scene reconstruction at L-band," in Proc. 7th EUSAR, Friedrichshafen, Germany, Jun. 2008, vol. 3

[36] M. D. A. D. ,. S. G. T. S. C. S. Howard Butler, «The GeoJSON Format Specification,» [En línea]. Available: http://geojson.org/geojson-spec.html. [Último acceso: 20 1 2016].

[37] D. Crockford, «Introducing JSON,» [En línea]. Available: http://www.json.org. [Último acceso: 20 1 2016].

Bibliografía

Criminisi, A., Reid, I., y Zisserman. *Single view metrology*. En: International Journal of Computer Vision. 2000. vol. 40, no. 2, p. 123–148.

Dragomir Anguelov, C. D. *GOOGLE STREET VIEW: CAPTURING THE WORLD AT STREET LEVEL.* En: IEEE Computer Society. 2010. vol. 43, no. 6, p. 32-38.

R. Hartley, A. Zisserman. *Multiple View Geometry in Computer Vision.* 2 ed, Cambridge University Press, 2004.

R. Szeliski, Computer Vision: *Algorithms and Applications*. 1 ed. Springer, 2010.