UNIVERSIDAD SANTO TOMÁS

Proyecto

Sistema de Reconstrucción de Mosaicos con Fusión de Información de Profundidad

Desarrollado por:

Gilberth Steeven Montejo Díaz

Dirigido por:

Ing. Edgar Andrés Gutiérrez Cáceres Esp. Msc. Phd. (st)

Ing. Camilo Ernesto Pardo Beainy Esp. Msc. Phd. (c)

Ing. William Fabian Chaparro Becerra Esp. Msc. Phd.

2022

_____

Gilberth Steeven Montejo Diaz

Notas de Aceptación

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Firma del director

_____

Firma del codirector

_____

Firma del Jurado

_____

Firma del Jurado

Dedicado a mi familia por su apoyo incondicional durante mi carrera y en especial a mi abuela; también para todos aquellos que fueron de mi proceso de formación y por supuesto, me dedico este trabajo, que consolida uno de los primeros grandes logros de mi vida.

Gilberth Steeven Montejo Diaz

Agradecimientos

Quiero expresar agradecimiento a mi familia por ser parte fundamental y motivación para hoy estar donde estoy.

Un agradecimiento especial a mis compañeros, por ser parte fundamental en el desarrollo profesional y por su buen ambiente que ha hecho de estos años de carrera, algunos de los mejores de mi vida.

También, quiero agradecer a los docentes, que de alguna u otra forma aportaron a mi crecimiento profesional con su conocimiento y apoyo.

Contenido

Resumen 9

1. Introducción 10

2. Presentación del Problema 11

3. Justificación 12

4. Objetivos 13

4.1. Objetivo General 13

4.2. Objetivos Específicos 13

5. Marco Referencial 14

5.1. Marco Teórico 14

5.1.1. Fotogrametría 14

5.1.2. Kinect 14

5.1.3. Color en las Nubes de Puntos} 15

5.1.4. Teoría para superponer imágenes obtenidas por el kinect V1 15

5.2. Estado del Arte 16

6. Análisis del Problema y Propuesta de Necesidades a suplir con el Software 19

6.1. Fusión de Información 19

6.1.1. Reconstrucción de la Superficie 19

6.1.2. Superposición de Imágenes RGB 20

6.1.3. Superposición de mapas de profundidad 21

6.2. Resumen: Fusión de Información 21

6.3. Nubes de Puntos y Mesh 22

6.3.1. Nubes de Puntos 22

6.3.2. Mesh o Malla 24

6.3.3. Resumen: Nubes de Puntos y Mesh 24

6.4. Interfaz Gráfica 25
6.4.1. GUI: Adquisición de Datos 25
6.4.2. Requisitos de Interfaz 25
6.4.3. GUI: Procesamiento de Datos 25
6.4.4. Resumen: GUI 26
7. Diseño de la Interfaz y Estructura Fundamental del Código del Software 27
7.1. Software y Hardware 27
7.1.1. Instalación de Freenect 28
7.2. Funcionamiento de la Aplicación 30
7.2.1. Interfaz Principal 30
7.2.2. Adquisición de datos 30
7.2.3. Generación de Modelos 3D 32
7.3. Metodología de desarrollo 34
8. Resultados 36
8.1. GUI Desarrollada 36
8.2. GUI: Adquisición de Datos 37
8.3. GUI: Procesamiento de los Datos 38
9. Conclusiones 41
9.1. Problemas Encontrados 41
9.2. Proyectos Futuros 41
9.3. Conclusiones Finales 42
9.4. Estrategias de Divulgación 44
9.4.1. Certificados 44
Referencias 45
Anexos 48

Anexo 1: Certificado de Registro de Software 48

Anexo 2: Manual de Usuario 48

Anexo 3: Artículo 48

Resumen

Se ha desarrollado una aplicación para el sistema operativo Ubuntu 18.04 en lenguaje Python 3.6, que permite la reconstrucción de mosaicos con fusión de información de profundidad, El software cuenta con una interfaz que permite el lanzamiento de dos subsistemas. El primer subsistema se encarga de establecer una conexión con el Kinect V1, (que es el dispositivo de adquisición de datos utilizado) y permite la captura y almacenamiento de muestras de una escena. El segundo subsistema se encarga de procesar los datos e integra un filtro de distancia, que permite seleccionar o excluir objetos de interés. El resultado final se obtiene en formato ply y se crea a partir del procesamiento de una nube de puntos, con la información adquirida.

1. Introducción

El uso de tecnologías para la reconstrucción de entornos se ha desarrollado en diferentes áreas de la ciencia, para poder obtener datos manipulables de escenas de interés. Dentro de este proyecto, se ha buscado generar un aplicativo con fines generales de aplicación, mediante la captura y procesamiento de datos obtenidos a partir de un Kinect V1; con esto también se ha considerado que, los modelos finales cuenten con medidas de profundidad en centímetros, esto con el fin de ofrecer al usuario final, una herramienta que permita la generación de modelos, similares a la escena del espacio real.

Con el desarrollo de este proyecto, se implementa un sistema de reconstrucción de mosaicos con fusión de información de profundidad, con una interfaz gráfica y dos subsistemas: uno para la toma de datos y otro para el procesamiento de los mismos. Dentro del procesamiento de los datos, se incluye un filtro de distancia, que permite seleccionar o excluir objetos de interés a partir del ingreso de distancias en centímetros, y siempre pensando em el procesado rápido de señales.

2. Presentación del Problema

Los mosaicos de imágenes son utilizados en múltiples campos de la ciencia [1]; dentro de ellos es posible encontrar la agricultura, la cartografía, la arquitectura y la ingeniería. Sin embargo, es de destacar que, en general, los mosaicos que se generan son imágenes que contienen la información del color de la escena y se genera sobre un plano bidimensional.

Al tener mosaicos 2-D, se desprecia la información de la profundidad de la escena, y en los casos que esta es requerida, se utilizan otros sistemas para determinar la misma. Por otro lado, para analizar las escenas, conocer la distancia de los objetos respecto al dispositivo de captura, ayuda a generar filtros a partir de la posición de los objetos en el espacio y a discernir de una manera óptima la información que se esté buscando en el momento [2].

El sistema de reconstrucción de mosaicos con fusión de información de profundidad se limita a la generación de una aplicación con interfaz gráfica de usuario mediante la cual se provea de opciones y herramientas para reconstruir una escena, mostrar lo reconstruido, orientar en coordenadas 3D el resultado generado, filtrar objetos a partir de distancias ajustables.

Para la reconstrucción de la escena se deberá tomar muestras de la información previamente y desde la interfaz se proveerá de un apartado para cargar los datos tomados. En cuanto a las demás opciones mencionadas anteriormente, se podrán ejecutar una vez reconstruido el entorno.

Desde la interfaz se podrán tomar los datos, pero se limita a la utilización del sensor Kinect V1. Los formatos aceptados serán .jpg para el caso de los datos RGB y .csv para la información de profundidad; esto se plantea a partir de los tipos de datos generados por el kinect v1, sobre el cual se generarán todas las pruebas del sistema planteado en este proyecto. Las medidas de altura y ancho de la escena conservarán su característico formato en píxeles, mientras que para la profundidad si se tendrá un formato en centímetros.

3. Justificación

La fotogrametría es una técnica que permite realizar mediciones y levantar modelos panorámicos de una escena y que, con los avances tecnológicos y desarrollos de algoritmos de visión artificial, ha evolucionado para incursionar en el campo de la reconstrucción tridimensional. Este proyecto se genera porque los modelos 3D tradicionales se generan con profundidades relativas en píxeles y, mediante trasformaciones se estiman valores equivalentes en medidas de mundo real. Para llevar a cabo ello, se utilizan sistemas de alto coste, generalmente incorporando sensores LIDAR [3] [4] y en ocasiones despreciando el color de las escenas.

Este sistema se plantea para ofrecer una aplicación de bajo coste que genere mosaicos con información de profundidad y color de escena y que sea aplicable con un único dispositivo a partir de la fusión de mapas de profundidad e imágenes RGB. También, para ofrecer una aplicación de propósito general, esta solucionará los problemas de filtrado por distancia y la visualización de objetos en un rango determinado por el usuario mediante opciones de configuración de parámetros de distancia para cada escena.

4. Objetivos

4.1. Objetivo General

Desarrollar una aplicación que permita la reconstrucción y visualización de mosaicos con información de profundidad, a partir de imágenes RGB y mapas de profundidad extraídos del Kinect V1.

4.2. Objetivos Específicos

- Emplear un filtro para los datos obtenidos y así evitar tiempos largos de procesamiento.
- Construir una nube de puntos a partir de imágenes RGB y datos de un mapa de profundidad.
- Crear un Mesh a partir de la nube de puntos generado.
- Desarrollar una interfaz gráfica de Usuario que facilite el uso del sistema.
- Diseñar un filtro de distancia desde la interfaz gráfica que permita seleccionar o excluir objetos en un rango determinado por el usuario

5. Marco Referencial

5.1. Marco Teórico

5.1.1. Fotogrametría

La fotogrametría es una técnica que permite realizar medidas 3D y generar volúmenes a partir de imágenes, y que, ha evolucionado con la tecnología y el desarrollo de algoritmos de visión artificial hasta llegar a incursionar en el campo de la reconstrucción 3D [5]. Para el caso último mencionado, se utilizan técnicas como la búsqueda de características similares en imágenes y el posicionamiento de cámaras mediante algoritmos, sin embargo, para este proyecto, se toma únicamente los píxeles centrales de la imagen RGB y sus correspondientes en el mapa de profundidad y se ubican de manera horizontal.

5.1.2. Kinect

El kinect es un dispositivo de bajo costo desarrollado por Microsoft, inicialmente enfocado para sus consolas Xbox y luego llevado a compatibilidad con computadoras de escritorio. Este sistema puede capturar imágenes RGB y mapas de profundidad de alta resolución [6], [7].

Este sistema se divide en dos partes: un láser emisor de luz Infrarroja (IR) y la cámara de luz infrarroja. Para la detección de profundidad se utilizan patrones de luz infrarroja emitidas por el proyector IR; la reflexión de este patrón es captada por la cámara IR. Los datos capturados son trasmitidos a un chip que se encarga de traducirlos y generar la imagen de profundidad. El alcance de este sistema es de 58° en horizontal, 45° en vertical y 70° en diagonal con un rango de operación de entre 0.8m y 3.5m [8]. Para evitar que la luz del ambiente tenga influencia, dado que el emisor es de una longitud de onda exacta, entonces, existe un filtro para las longitudes de onda de 830nm sobre la cámara IR. El problema principal con el Kinect en exteriores, se halla en que, el rango de frecuencias de la luz solar es amplio en el espectro de infrarrojo. La distancia es limitada por el diodo que se utiliza como sistema de emisión de ondas.

Las aplicaciones de este dispositivo se centran en la visión artificial, y para la aplicación actual, se trabaja sobre el sistema de libre distribución Ubuntu y como driver la librería libfreenect [9].

5.1.3. Color en las Nubes de Puntos}

Para obtener el color en las nubes de puntos basta con habilitar la imagen RGB y obtener la equivalencia de la nube respecto a la imagen RGB. El problema se basa en que la imagen RGB no coincide con la nube de puntos debido que no están ubicadas en el mismo lugar, por lo tanto, las imágenes capturadas están ligeramente desviadas. Sin embargo, al utilizar la librería de freenect, este problema es corregido.

5.1.4. Teoría para superponer imágenes obtenidas por el kinect V1

Para poder determinar la cantidad de datos que se deben de cada una de las imágenes RGB se analiza el sistema de adquisición de datos ubicado en el espacio tal y como se muestra en Fig. 1. En el modelo dispuesto se tiene el sistema coordenado (x, y) y un distancia d desde el foco del sensor hasta el punto de interés. El ángulo α indica la inclinación de la línea d respecto al eje coordenado y.

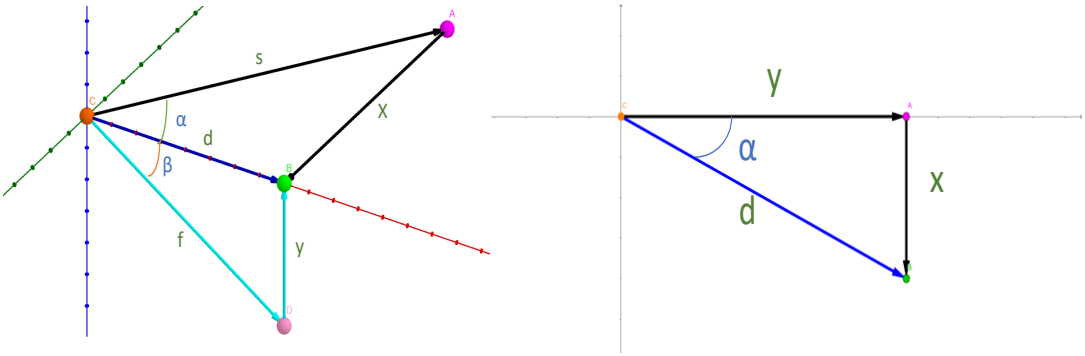

Fig. 1. Modelo de adquisición de datos ubicado en el espacio

Para determinar la ubicación del pixel, se debe tener en cuenta que el máximo ángulo horizontal es de 57°, lo que quiere decir que para una resolución de 640gpx se tienen 0.089°/px; en cuanto al máximo ángulo vertical, este es de 43°, de manera que para una

resolución de 480px, corresponden 0.0896°/px. Sabiendo esto, y asumiendo que el punto B es el centro de la imagen (Ver Fig. 1), entonces:

$$B = (320,240)$$

De tal manera que el ángulo β viene dado por la distancia entre B_x y A_x y multiplicado por la resolución vertical correspondiente a cada pixel. Para el caso de α funciona de igual manera, pero será la distancia entre B_y y A_y y la resolución horizontal. En ecuaciones, se tiene:

$$\beta = |B_x - A_x| * 0.089°/px \quad (1)$$

$$\alpha = |B_y - A_y| * 0.0896°/px \quad (2)$$

Por otro lado, se tiene un valor λ; la forma de calcular este se da dentro de la librería de libfreenect:

$$\lambda = \frac{1}{(-3.07 * 10^{-3} * ValorKinect + 3.33)} \quad (3)$$

Y esta ecuación sirve para convertir a metros los valores de profundidad medidos por el sensor de profundidad del Kinect. Esta ecuación se limita al rango de valores de 500 a 1023 entregados por el dispositivo. Ahora sí, retornando al modelo propuesto en la Fig. 1, se determinan las ecuaciones para el mismo:

$$d = \lambda \cos(\beta) \quad (4)$$

$$x = d\sin(\alpha) \quad (5)$$

$$y = d\cos(\alpha) \quad (6)$$

Para las ecuaciones (4), (5), (6) los valores se obtienen en metros.

5.2. Estado del Arte

Los trabajos de investigación en el campo de la reconstrucción tridimensional son de aplicaciones generales y evidencian la aplicación de la visión artificial en los diferentes campos del conocimiento, como la agricultura en donde una de sus posibles aplicaciones se basa en la reconstrucción de malas hierbas para un análisis geométrico de las mismas [10]; en campos como la arqueología y sus derivados eventos turísticos [5] sirve como referencia para ofrecer mejores experiencias.

Para las aplicaciones mencionadas anteriormente, se parte de fotografías y mediante el posicionamiento de cámaras [11] se generan nubes de puntos representativos de una escena, que posteriormente son utilizados para generar las mallas correspondientes [12]. El modelo generado se puede exportar a diferentes formatos y plataformas, y haciendo uso de las propiedades del procesador de Texto LaTEX, la generación de documentos interactivos con modelos 3D en formato Universal 3d [13]. Es de resaltar que los resultados finales tienen medidas en píxeles.

En aplicaciones del kinect como sensor, se utilizan sistemas muy similares al anteriormente mencionado, pero diferenciado en los resultados finales por ofrecer resultados con medidas en milímetros tal y como se hace en [6], [14], [15]; bajo el mismo sensor se desarrollan técnicas de ubicación de objetos en el espacio y determinación de distancias a partir del mapa de profundidad.

Para la reconstrucción de entornos en tres dimensiones existen diferentes métodos, sin embargo, todas se reducen en el mismo principio: medir la distancia desde el sensor hasta el punto observado; esta medición genera puntos 3D de posición. Los datos obtenidos también pueden denominarse como nube de puntos [16]. Un aspecto que se debe tener muy en cuenta es las condiciones del entorno como la iluminación, el ángulo de captura y la velocidad de captura, pues estos, pueden afectar los resultados finales.

Existen dos técnicas de reconstrucción: reconstrucción pasiva y activa. En cuanto a la técnica de reconstrucción pasiva se parte de escáneres que devuelven información en forma de puntos 3D. Dentro de estos tipos de escáneres es posible encontrar los LIDAR y los Sensores de Profundidad. Para la técnica de reconstrucción pasiva se parte de un conjunto de imágenes (videos incluidos) y mediante la estimación de la posición de cámaras, se establece y generan los modelos de puntos 3D. También existe un tercer método, aunque menos desarrollado, y consiste en la combinación de las dos técnicas anteriormente mencionadas [17].

Algunos sistemas de reconstrucción de entornos pueden llegar a ser demasiado costosos, dentro de estos se encuentran los que involucran satélites. Cuando se involucran satélites las medidas del objeto a reconstruir se pueden estimar a partir de la longitud, la latitud y la altitud. Dentro de estas técnicas, es posible encontrar que se apliquen métodos diferentes a los mencionados anteriormente; estos se pueden basar en la consolación de pares de imágenes

y la estimación de gradientes de energía [18]. Con estos datos procesados, se analiza posteriormente el peso de cada uno de los resultados y mediante sistemas de Inteligencia artificial se determina si los puntos pertenecen o no al objeto que se está reconstruyendo.

Tal y cómo se mencionaba anteriormente, la inteligencia artificial también es aplicada en el campo de la reconstrucción de entornos y objetos tridimensionales; con esto, se da la introducción y tal y como hace [18], se utilizan redes neuronales profundas para la estimación de puntos pertenecientes a un modelo y además de ello para aumentar la precisión en la estimación de la profundidad asociada a cada punto de la nube de puntos generada [19].

6. Análisis del Problema y Propuesta de Necesidades a suplir con el Software

6.1. Fusión de Información

En esta sección se aborda el análisis realizado para llevar a cabo el desarrollo del proyecto; esto partiendo de las preguntas problémicas planteadas. Esta sección determina el camino sobre el cual se trabajará durante los siguientes capítulos.

6.1.1. Reconstrucción de la Superficie

Para la reconstrucción del mosaico con fusión de información de profundidad, se parte de dos tipos de datos capturados: el primero son imágenes RGB y el segundo tipo de datos es un mapa de profundidad. Para los dos casos se obtienen matrices de 640x480. Esto depende del dispositivo que se usa para capturar los datos; en el caso de este proyecto, el dispositivo empleado para la obtención de los datos es el Kinect V1.

Primero, es de mencionar, que, para el alcance de este proyecto, se asume que los dos sensores se encuentran en la misma posición. Teniendo en cuenta esto, si se captura un dato, y dado que el tamaño de los datos obtenidos para cada tipo es igual, el valor que ocupe la posición (n, m) del mapa de profundidades será correspondiente a la profundidad para el valor RGB en (n, m) de la imagen RGB, tal y como se observa en Fig. *2*.

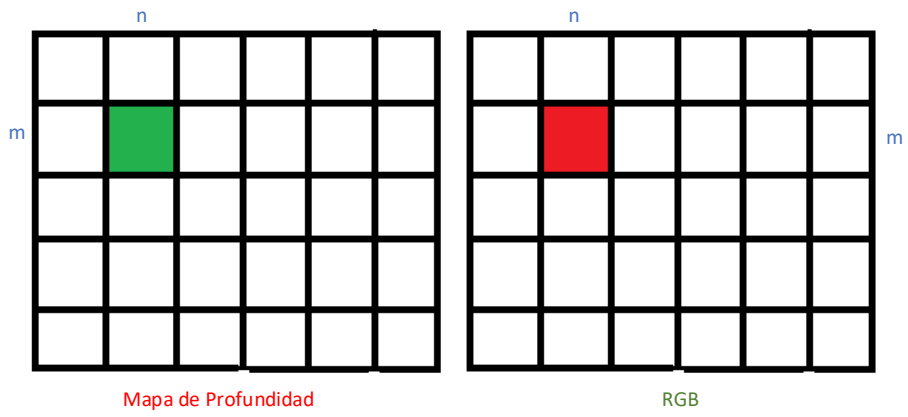

Fig. 2. El punto (n, m) del mapa de profundidad corresponde a la profundidad del punto (n, m) en RGB

En la sección 5.1.4 se muestra el análisis de la posición del Kinect respecto a una imagen. Entonces, partiendo de las ecuaciones planteadas en el capítulo mencionado, se tiene un Angulo α que hace referencia a la distancia horizontal entre dos puntos, y un valor x que referencia a esta distancia. La ecuación (5) relaciona estas dos variables, sin embargo, el valor que se desea despejar es el valor de α dado que es la cantidad de pixeles equivalentes a un valor de distancia dado del mundo real. La idea principal de obtener este valor radica en el hecho de que se tienen varias imágenes.

6.1.2. Superposición de Imágenes RGB

Es necesario generar la superposición de las imágenes dado que se tiene una serie de muestras que caracterizan la escena. Para la toma de estos datos se requiere que la distancia entre muestra y muestra sea un valor constante; para el análisis del sistema partimos de lo que se propone en la Fig. 3 en donde se observa que las imágenes comparten cierta información. La idea principal para generar los modelos, es eliminar de la imagen dos, los datos que comparte con la imagen uno. Esto se plantea para dar poder ofrecer tiempos cortos de procesamiento de datos para obtener un modelo tridimensional final.

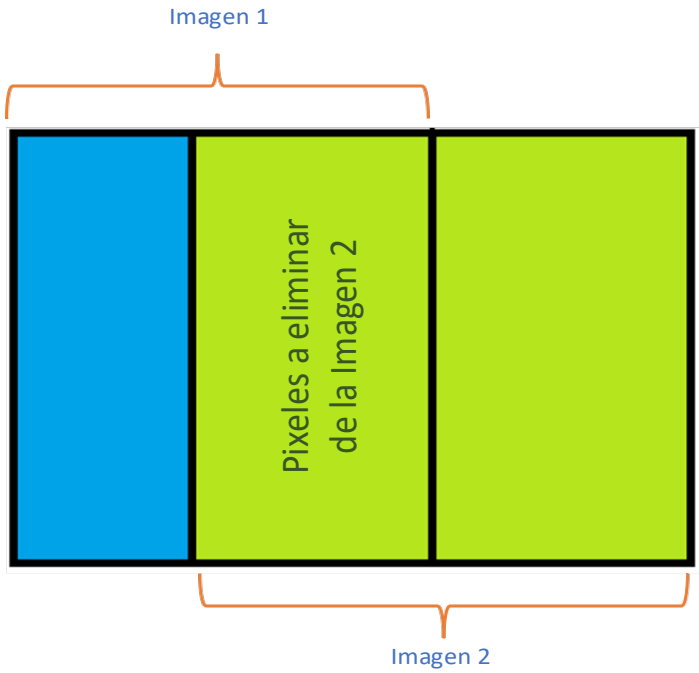

Fig. 3. Modelo de Superposición de Imágenes

Para poder determinar la cantidad de pixeles que se deben conservar, se hace uso de la ecuación (5) de donde vamos a despejar α:

$$\alpha = \sin^{-1}\left(\frac{x}{d}\right) \tag{7}$$

De (7) es conocido el valor de x, y hace referencia al valor de la distancia entre las tomas de datos, y reemplazando el valor de d en la ecuación (7):

$$\alpha = \sin^{-1}\left(\frac{x}{\lambda \cos(\beta)}\right) \tag{8}$$

Dado que se quieren conservar toda la información vertical, el valor de β siempre se puede aproximar a 42. Por otro lado, el resultado de esta ecuación estará dada en grados y se necesita expresar un valor en pixeles. Para poder llevar esto de grados a pixeles, se debe dividir por 0.089° el resultado; según la ecuación (2). Y el valor obtenido será la cantidad de pixeles que se deben conservar de la segunda imagen. La ecuación final será:

$$\#\ de\ pixeles\ a\ conservar = \frac{\sin^{-1}\left(\frac{x}{\lambda \cos(\beta)}\right)}{0.089°} \tag{9}$$

6.1.3. Superposición de mapas de profundidad

Anteriormente se ha mencionado que un valor (n, m) de un mapa de profundidad describe la profundidad del pixel (n, m) de la imagen RGB. Haciendo uso de esta aproximación y teniendo en cuenta lo expuesto en la subsección anterior; la cantidad de pixeles a conservar de la imagen dos, es la misma cantidad de posiciones a conservar de la matriz de profundidad correspondiente a la misma imagen.

De esta manera, se da solución a la fusión de información de color y profundidad para el sistema de reconstrucción de mosaicos con fusión de información de profundidad.

6.2. Resumen: Fusión de Información

Para sintetizar lo expuesto a lo largo de esta sección, para el sistema que se desarrolla en este documento:

- La distancia entre las muestras de un mismo proyecto, debe ser siempre la misma.

- Se asume que el punto desde el cual fue capturada la información RGB de la escena, es exactamente el mismo desde el cual se capturan los datos de profundidad.
- Al asumir lo expuesto en el elemento anterior, se puede considerar que un punto (n, m) del mapa de profundidad representa la profundidad del mismo punto, pero en la matriz de colores. (Ver figura Fig. 3)
- La superposición de las imágenes se da a partir de la eliminación de los pixeles comunes entre dos diferentes muestras. Aplica de igual manera para el mapa de profundidad, sin embargo, los cálculos se realizan sobre la información de color, dado que esta tiene un formato de pixeles y se pueden convertir a valores de distancia horizontal en el mundo real.
- Al basar la teoría de procesamiento en la eliminación de datos, el tiempo de procesamiento y el costo computacional es bajo, pues el proceso central se basa en operaciones matemáticas únicamente. Esto se explica más adelante.

6.3. Nubes de Puntos y Mesh

Los puntos de una nube de puntos se representan, generalmente en el espacio y contienen información puntual del tipo (x,y,z); para el modelo que se analiza en este documento, el valor de z hace referencia a la profundidad del punto en la escena, es decir, esta información se extrae del mapa de profundidad. Para los otros dos valores, son las coordenadas en pixeles que se obtienen a partir de la información tricromática entregada por el dispositivo Kinect.

6.3.1. Nubes de Puntos

A simple vista, obtener una nube de puntos, a partir de lo planteado en el capítulo anterior, podría parecer algo bastante complicado, sin embargo, no es así. Una imagen RGB es una matriz en donde cada una de las **posiciones** es de la forma [R, G, B]. En los mapas de profundidad se tiene una matriz donde cada una de las **posiciones** tiene la forma [Valor de profundidad en metros]. En el capítulo anterior, se plantea la forma de fusionar los datos de color, y los datos de profundidad y se plantea un modelo que, devuelve como resultado dos matrices de igual manera; estas matrices ya no serán de 640x480 (tamaño de las matrices de

información que entrega el Kinect V1) sino que tendrán un tamaño de nx480. En donde n es un valor que cambia para cada escena, y depende de que tantas muestras se toman de la escena y la distancia entre ellas.

Es de recordar que la cantidad de pixeles en vertical no cambia dado que el Kinect solo se desplaza de manera horizontal.

Ahora, la diferencia entre obtener una imagen en 2D y obtener una nube de puntos radica en la fusión de la información y en la forma en cómo la misma se dispone sobre el plano. La nube de puntos realmente existe desde el momento en que se captura cualquiera de los dos tipos de datos que se obtienen del Kinect. Sin embargo, con el sólo mapa de profundidad, se obtienen puntos en el espacio, pero sin color, y con la imagen RGB se obtienen puntos de color, pero sin profundidad, es decir, todos alineados en el plano.

Entonces, siguiendo estas ideas: primero, la nube de puntos existe, pero para observarla de manera gráfica se debe graficar cada uno de estos valores como punto y no todos como imagen. En segundo lugar, para obtener una nube de puntos con colores y profundidad, al momento de graficar se debe disponer en el grafico en el espacio del valor (n, m) del resultado de imágenes superpuestas analizado en el capítulo anterior y de los mapas de profundidad superpuestos.

El resultado final se puede analizar como la organización de prismas rectangulares con caras iguales (ver figura Fig. 4), en donde (n, m) es la posición en la matriz, RGB es el color del punto, y P es el valor de profundidad de la figura.

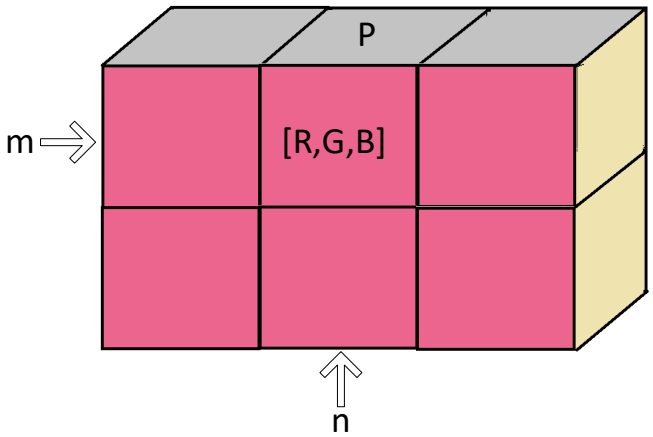

Fig. 4. Modelo de nube de puntos representado como prismas rectangulares con caras iguales

6.3.2. Mesh o Malla

No se ha comentado anteriormente, pero este proyecto se ha desarrollado en el lenguaje de programación Python; esto se hace relevante en esta sección dado que para generar el mesh se utiliza una librería específica de este lenguaje y con ello, el análisis se fundamenta en los pasos que la misma requiere. 344

Inicialmente, se retoma lo comentado en la sección 6.3.1; una nube de puntos que contiene la información de color, y una nube de puntos que contiene la información de profundidad. Las dos nubes de puntos tienen un tamaño de nx480. Con estos datos se procede a crear los vóxel, que tal y como se comentaba en Capítulos anteriores, son el equivalente a los pixeles de 2D en 3D. El vóxel agrupa en una sola matriz la información de profundidad y de color. Con los datos agrupados, ahora se debe estimar la normal a cada vóxel generado. Esto se hace, debido a que es requerido para realizar la reconstrucción de superficie de Poisson que aplica el algoritmo de generación de malla triangular. (agregar referencia de este)

6.3.3. Resumen: Nubes de Puntos y Mesh

En esta sección se abordan los temas referentes a la manipulación de nubes de puntos y cómo pasar de la nube de puntos al mesh. De esta sección se destaca:

- A nivel de gráfico, para observar la nube de puntos, se debe graficar cada punto de la imagen RGB por separado y no como imagen.
- La nube de puntos existe, desde que se capturan los datos, sin embargo, se debe tener cuenta el ítem anterior.
- Se generan dos nubes de puntos, una contiene datos de color, y los otros datos de profundidad. Al graficarlas juntas, se obtienen puntos con color y profundidad.
- Para generar el Mesh, el primer paso es crear una matriz de pixeles volumétricos que contiene la información de color y profundidad.
- A partir de la matriz de pixeles volumétricos, se estima la normal para cada vóxel.
- Mediante el método de reconstrucción de superficies de Poisson, se obtiene el modelo tridimensional final.

6.4. Interfaz Gráfica

La interfaz gráfica es un factor fundamental en cualquier software, pues con esta se le otorga el factor amigable con el usuario. En este proyecto y de la forma en cómo se ha analizado, se hace pertinente el desarrollo de una interfaz gráfica con dos aspectos fundamentales: el primero considera la adquisición de datos y el segundo incluye la generación de los modelos finales.

6.4.1. GUI: Adquisición de Datos

Para la adquisición de los datos, se debe tener en cuenta que esta depende de un dispositivo de adquisición de datos: Kinect V1. Para el correcto funcionamiento del dispositivo de obtención de valores, es requerida la utilización de drivers. El driver y librería que se utiliza para lograr la conexión del Kinect con Ubuntu 18.04 se denomina **Freenect.** Esta librería es de uso libre. No solamente se requiere de la instalación de la misma, se requiere de importar el wrapper de Python, para poder controlar el flujo y tratamiento de datos del dispositivo al computador desde este lenguaje.

6.4.2. Requisitos de Interfaz

Habiendo solucionado la conexión del Kinect, el sistema se encuentra en las condiciones necesarias para pensar en lo que se requiere para la GUI. En un primer aspecto, desde la GUI se debe incluir un indicador del estado de conexión del dispositivo de adquisición de datos; además de proveer entradas para seleccionar nombre del proyecto, y la ruta en donde se desea guardar y un método para la captura manual de cada una de las escenas.

6.4.3. GUI: Procesamiento de Datos

El procesamiento de los datos tiene un enfoque más permisible que la adquisición de datos. Sin embargo, se debe tener en cuenta que, los datos capturados mediante la interfaz descrita en 6.1, tienen nombres clave: depthxx.csv para los mapas de profundidad y imgxx.jpg para las imágenes RGB. El valor de xx son números y deben ser consecutivos,

dado que se pueden procesar infinitas imágenes, siempre y cuando cumplan estas condiciones.

Siguiendo el orden de ideas planteado, la interfaz de procesamiento de datos debe permitir seleccionar la ruta de origen de los archivos y la ruta donde se guarda el modelo 3D generado. Adicional a esto, debe solicitar el ingreso de un nombre para el proyecto y la distancia (en centímetros) entre muestra y muestra. Además de ello, se incluye una opción de agregar un filtro de distancia (en centímetros), que debe ser opcional. Esta opción de filtro de distancia tiene como objetivo permitir al usuario manipular el modelo final, de tal manera que pueda excluir objetos en un rango de distancia especificado.

6.4.4. Resumen: GUI

La interfaz gráfica es parte fundamental de un software, pues esta otorga hace que el uso del sistema sea más fácil. Para esta aplicación la interfaz gráfica se divide en dos acciones: la primera se enfoca en la adquisición de datos, y la segunda en el procesamiento de datos. Para la interfaz de adquisición de datos se incluye:

- Instalación de Drivers para el funcionamiento del Kinect
- Posibilidad de que el usuario final pueda elegir la ruta en donde se van a guardar los datos adquiridos.
- Nombrar el proyecto
- Botón para capturar las muestras de manera manual
- Indicador de estado actual del Kinect V1.

En cuanto a la interfaz de procesamiento de datos, se incluyen las funcionalidades de:

- Seleccionar la ruta de origen de los datos a procesar. Estos datos deben estar nombrados de la siguiente manera: depthxx.csv y imgxx.jpg para mapas de profundidad e imágenes RGB respectivamente. El valor de xx es un número y debe estar de manera accedente o el sistema no funcionará correctamente.
- Ingresar el nombre del modelo 3D a generar.
- Ingresar la distancia en centímetros que existe entre cada una de las muestras.

7. Diseño de la Interfaz y Estructura Fundamental del Código del Software

En este capítulo se busca explicar de manera detallada la aplicación y su funcionamiento sin recurrir a la mención de códigos; también se establecen los límites del software desarrollado.

7.1. Software y Hardware

Esta aplicación está desarrollada para el sistema operativo Ubuntu 18.04 y ha sido codificada en lenguaje Python 3.6, además de ello, la forma de obtener los datos es mediante un Kinect V1. Teniendo en cuenta estas consideraciones de software y hardware se generan algunas limitantes para la ejecución y lanzamiento del aplicativo sobre el sistema operativo mencionado.

Iniciando por el diseño del software se hace necesario tener en cuenta la versión de Python que se tiene instalada y con ello una serie de librerías que son utilizadas a lo largo de la ejecución de la aplicación; estas son mencionadas en la siguiente sección. Se decide trabajar sobre un sistema operativo basado en Linux dada la facilidad para manipular los datos que se adquieren a través del Kinect. Es de destacar, que el dispositivo es fabricado por la empresa Microsoft y la misma ofrece aplicaciones de uso como el SDK de Kinect que se encarga de instalar los drivers y proveer de un entorno para el correcto funcionamiento del dispositivo; sin embargo, estos están disponibles para el sistema operativo de la compañía fabricante.

El principal factor para trabajar sobre Linux, radica en la posibilidad de manipular los datos, tal y como se ha mencionado anteriormente. Pero, ¿a qué se hace referencia con manipular los datos? Cuando se hace mención de esto, se hace referencia a la posibilidad de tener documentación de uso libre, y un soporte amplio para la librería que se encarga de traducir los datos que el sensor captura. También se analiza en este aspecto, la forma más adecuada para la manipulación de los datos finales y esto incrustado teniendo en cuenta los requerimientos de interfaz gráfica planteados en el capítulo anterior, y enfocados en las herramientas que pueda proveer un lenguaje, de cara a la manipulación de los modelos generados.

Habiendo explicado las razones por las cuales se ha elegido trabajar sobre el sistema operativo, también se hace necesario justificar el dispositivo de adquisición de datos utilizado. Al hablar de mosaicos, es necesario incluir el término de imágenes; y al incluir el término de "profundidad" se incluye otro requerimiento de sensor en el proyecto. En este punto, hay dos opciones, utilizar un dispositivo que satisfaga los dos requerimientos de sensor, o buscar satisfacer las necesidades con sensores diferentes y antes de continuar, también es necesario comentar, que la solución debe ser barata.

Actualmente en el mercado existen diversos sensores que pueden capturar mapas de profundidad, dentro de ellos, se encuentran los LIDAR y los sistemas infrarrojos. Por un lado, los primeros mencionados pueden llegar a tener costos elevados, dependiendo de la tecnología que usen, y tienen la limitante de la luz. En el caso de los sensores por infrarrojo, la luz es un problema, pero no afecta de igual manera, en como pasaría con el otro sensor. Por otro lado, los sensores que entreguen datos RGB, existen muchos más.

Si se utilizan sensores por separado, la resolución (en cuanto a tamaño del vector o cantidad de datos que devuelven por muestra) puede ser diferente y podría generar incongruencias en los modelos finales. Aquí es el punto en donde se elige el Kinect, pues este dispositivo captura datos RGB de una escena y mediante un sensor infrarrojo, devuelve mapas de profundidad. El nuevo problema radicaba en utilizar la primera o segunda versión del dispositivo; finalmente se decide utilizar la primera versión pues devuelve datos de un mayor tamaño, lo que indica que para un escenario se requieren menos muestras; sin embargo, este dispositivo no hace el ajuste de datos necesario para los sensores estéreo.

Para el funcionamiento del Kinect V1 se requiere de un sistema de controladores y existen diferentes opciones, algunas que, a la fecha, no son de uso libre, otros son únicamente para un sistema operativo en específico; y otros, como Freenect, son de open source y están disponible con soporte para Ubuntu. A continuación, se describe de manera detallada el paso a paso para la instalación del Driver Freenect y el wrapper de Python para el mismo.

7.1.1. Instalación de Freenect

Para la instalación de los driver y librería para Ubuntu 18.04 y Python 3, se deben tener en cuenta que el computador debe contar con los siguientes paquetes instalados:

- Libusb >= 1.0.18
- -CMake >= 3.12.4
- Python 3 (también funciona en Python 2)
- OpenGL
- Glut

Para que el funcionamiento de libfreenect como librería de Python sea correcto, es requerido tener instalado los paquetes de uso libre:

- Cython
- Numpy
- Matplotlib
- Opencv-ṕython
- Pandas

Con estos requisitos validados en el computador, se procede a la instalación del driver. Para ello es necesario seguir los siguientes pasos, desde una terminal de Ubuntu:

```
git clone https://github.com/OpenKinect/libfreenect
cd libfreenect
mkdir build
cd build
cmake -L .. # -L lists all the project options
make
sudo make install
```

Ahora, se debe cargar la configuración al sistema y añadir las nuevas variables al mismo, para ello se escribe en la consola:

```
sudo ldconfig /usr/local/lib64/
sudo adduser $USER video
sudo adduser $USER plugdev
```

Finalmente, se crean nuevas reglas en el sistema, para que pueda recibir los datos del dispositivo, para ello, en la consola se digita:

```
sudo nano /etc/udev/rules.d/51-kinect.rules
```

Entonces la ventana emergente que se abre, se deberá escribir lo siguiente y guardar la configuración:

```
# ATTR{product}=="Xbox NUI Motor"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02b0", MODE="0666"
# ATTR{product}=="Xbox NUI Audio"
```

```
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02ad", MODE="0666"
# ATTR{product}=="Xbox NUI Camera"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02ae", MODE="0666”
# Kinect for Windows
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02c2", MODE="0666"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02be", MODE="0666"
SUBSYSTEM=="usb", ATTR{idVendor}=="045e", ATTR{idProduct}=="02bf", MODE="0666"
```

La prueba inicial será escribir en la consola: “libfreenect-glview” si se abre una ventana como la siguiente, ya es posible iniciar el análisis de la interfaz gráfica para la toma de muestras. Es de mencionar, que como se debe incluir el wrapper de Python, nos dirigimos a la carpeta con el nombre que indica ello, y lo instalamos con el compilador de paquetes de Python. Como nota final: no se puede trabajar sobre entornos virtuales, siempre que se trabaja con el Kinect V1 y freenect, se deben ejecutar los scritps desde el base del sistema.

7.2. Funcionamiento de la Aplicación

Con los detalles establecidos en el análisis del sistema, se establecen los componentes con los que debe contar la aplicación. A continuación, se expone la forma en cómo funcionan los dos apartados de la aplicación.

7.2.1. Interfaz Principal

La interfaz principal se desarrolla con el fin de integrar las dos funcionalidades del software desarrollado y su función es netamente la de ejecutar la sección para la adquisición de datos, o la sección para generar los modelos 3D. Dentro de la interfaz principal se encuentra también un botón para salir de la aplicación de forma segura, y otro llamado “Configuración” desde el cual se puede cambiar el tamaño de la ventana del programa.

7.2.2. Adquisición de datos

Para la adquisición de datos es necesario tener conectado el Kinect; el estado de esta conexión se indica en la interfaz. El usuario debe seleccionar la ruta de destino de los archivos y un nombre para el proyecto. Por seguridad, y evitar que se puedan confundir los datos de dos proyectos diferentes, en una misma carpeta, no puede haber dos proyectos con el mismo nombre: esto se valida desde el programa, y en caso de encontrar un proyecto con el mismo

nombre del nuevo o no encontrar la ruta indicada, saldrá una alerta en ventana emergente, indicando el problema que existe.

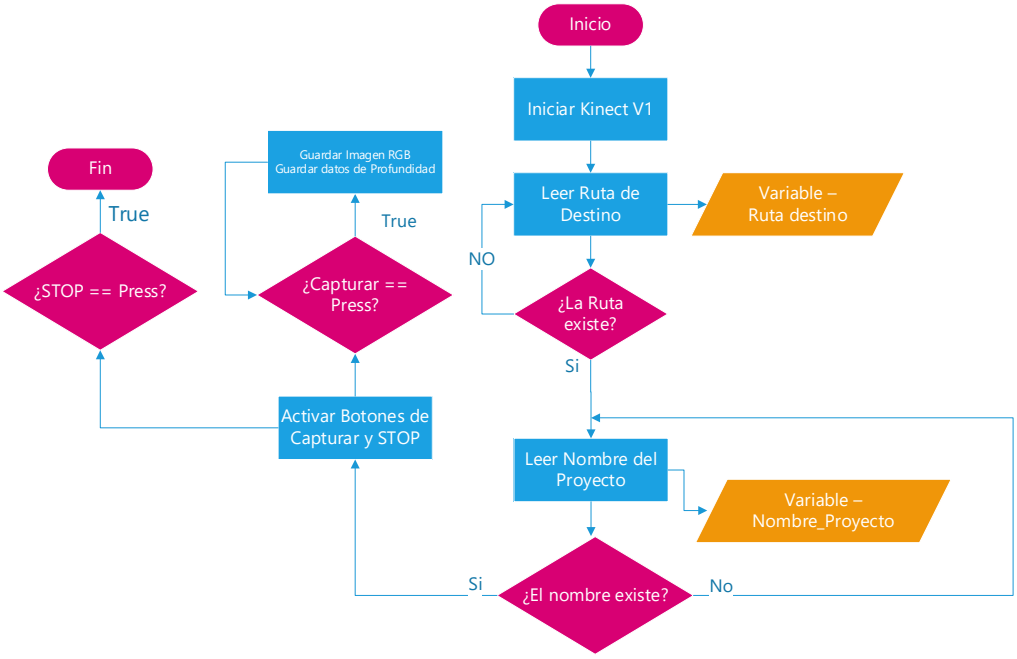

Fig. 5. Diagrama de Bloques para el sistema de adquisición de datos

Habiendo validado las rutas y el nombre, se activa la ventana que muestra lo que el sensor está enfocado, en tiempo real, y habilita un botón para realizar de manera manual, la captura de los datos (Fig. 5). Tal y como se indicaba en la fase de diseño, es necesario tomar las muestras de manera consecutiva y con un desplazamiento horizontal constante (ver Fig. 6). El formato para los archivos será JPG y CSV para las imágenes RGB y los mapas de profundidad respectivamente; y se almacenan directamente en una carpeta con el nombre y en la ruta indicados.

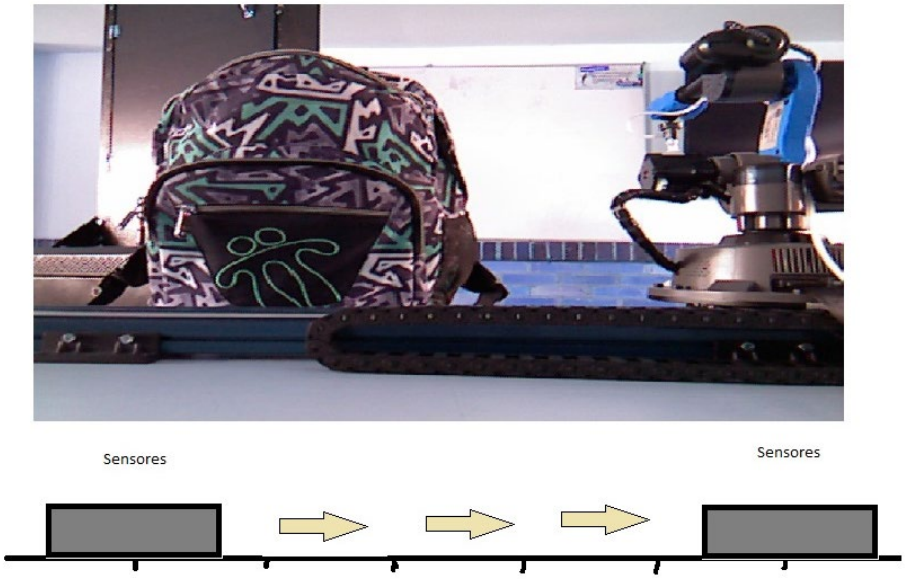

Fig. 6. Método para la adquisición de datos

Para el tratamiento de las imágenes se hace uso de la librería de OpenCV para Python3 y en cuanto al procesamiento de los datos de profundidad, el módulo encargado de ellos es Pandas.

7.2.3. Generación de Modelos 3D

Para obtener los mosaicos con fusión de profundidad (modelos 3D), es necesario aplicar diferentes técnicas para el procesamiento de los datos. Inicialmente, al programa (Fig. 7) se le debe proveer una ruta de origen, que hace referencia al directorio donde se encuentran las imágenes RGB y mapas de profundidad adquiridos mediante la interfaz expuesta en la sección anterior, y una ruta de destino, que será donde se va a guardar el modelo 3D final.

También se incluye un filtro de distancia, el cual permite ingresar una distancia mínima y una distancia máxima; estos dos valores sirven para realizar un filtrado en distancia a la escena, lo que permite incluir, excluir o enfocar objetos o detalles que se quieran analizar de la escena reconstruida. Si no se le dan valores al filtro, todos los puntos son procesados. También se debe incluir un nombre único de proyecto y la distancia a la cual fueron tomadas las muestras.

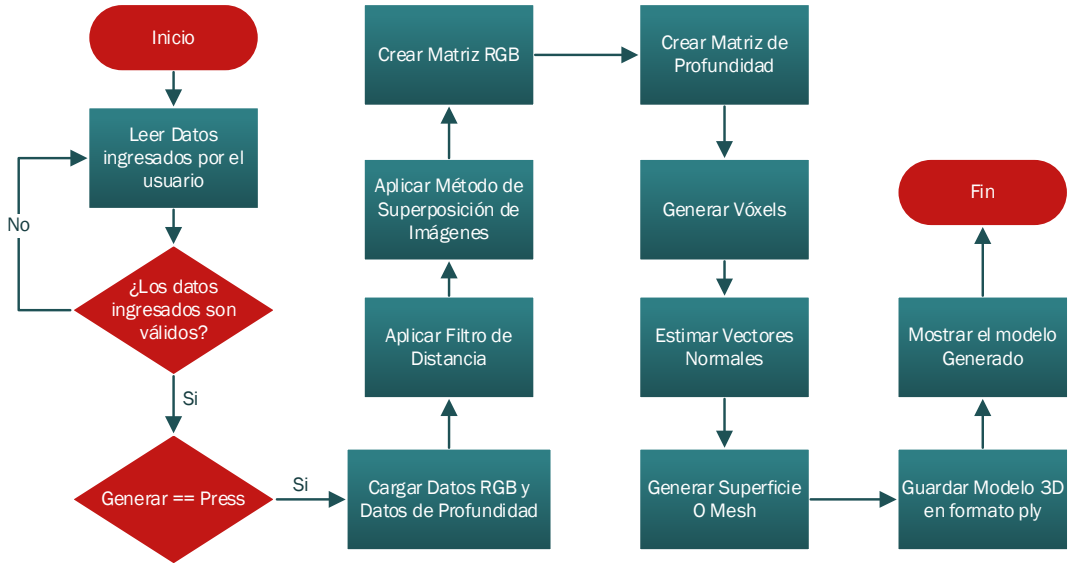

Fig. 7. Diagrama de Bloques para el sistema de Procesamiento de datos y generación del mosaico con fusión de información de Profundidad

Ahora bien, el funcionamiento a nivel código, solicita haber ingresado las rutas de origen y destino, lee los datos del filtro, el nombre del proyecto y la distancia a la cual se tomaron las muestras para el mosaico actual. Los primeros datos que se utilizan en la ejecución, son las rutas, en donde se verifican la cantidad de datos a importar y se importan, mientras esto sucede, se va aplicando el filtro de distancia ingresado (si no se ingresa, no se aplica) para posterior, realizar la aplicación del modelo de superposición de imágenes propuesto y analizado en el capítulo de análisis.

Al realizar primero el filtro, la nube de puntos tendrá menos puntos por procesar, y el tiempo de ejecución se disminuye, respecto al tiempo de procesamiento, con todos los datos. Es de aclarar, que no existe un tiempo estándar para el procesado de los datos, pues esto depende del tamaño y cantidad de muestras de la escena. Dado que este proyecto, no contempla la visualización de la nube de puntos, esto no se tiene en cuenta, y esta se utiliza únicamente como requisito para generar el mesh. Un último detalle, si el filtro elimina un punto de la nube de puntos de profundidad, también es eliminado su par, en la nube de puntos de color.

Teniendo la fusión de información, es decir, una nube de puntos representativa para color y otra para profundidad, se procede a crear los vóxel o pixeles volumétricos. La razón por la que, en los pasos anteriores se crean dos nubes de puntos, radica en que, para crear los vóxels,

mediante el módulo Open3D, es necesario tener la profundidad y el color en variables diferentes. Al agregar el color al vóxel, se debe dividir la variable por 255, pues el formato original de la imagen tiene valores para R, G y B entre 0 y 255, sin embargo, open3d requiere valores entre 0 y 1.

Teniendo los vóxels creados, se procede a la estimación de la normal a cada uno de ellos. Para esto se utiliza el método (incluido en open3d) *KDTreeSearchParamHybrid* que busca las normales en función de la estimación de un radio de superficie de los vóxels adyacentes. Con la estimación de normales terminada, el programa genera la superficie a partir del método *TriangleMesh* que genera el modelo 3D final a partir de una nube de puntos de Poisson. El formato del archivo final es *ply*.

7.3. Metodología de desarrollo

Para el desarrollo de la aplicación de este proyecto se utiliza el **modelo en cascada (**Fig. 8**)** que consiste en encadenar las diferentes etapas de la ejecución del proyecto y finalmente, establecer una retroalimentación del producto final. Este modelo se divide en cinco (5) pasos:

1. Análisis: Es la fase de preparación del proyecto y el objetivo es dar el punto de partida para el desarrollo del proyecto. En esta etapa se plantean las necesidades y los objetivos a alcanzar, para con ello generar un banco de requisitos que debe cumplir el software.
2. Diseño: Se plantean la estructura fundamental del código de la aplicación y con ello la organización de los elementos que debe contener el proyecto y cómo se relacionan entre sí. En esta etapa se debe tener en cuenta el diseño de la interfaz y siempre se debe pensar en el usuario final.
3. Implementación: En el diseño, se da un bosquejo de lo que se requiere, en esta etapa se define el lenguaje y se integra todo lo diseñado en un código.
4. Verificación: Comprobación y ejecución de todo el código final, y verificación de funcionamiento.
5. Mantenimiento: Analizar los resultados obtenidos, y si es necesario, realizar los cambios pertinentes y dar por terminado el proyecto.

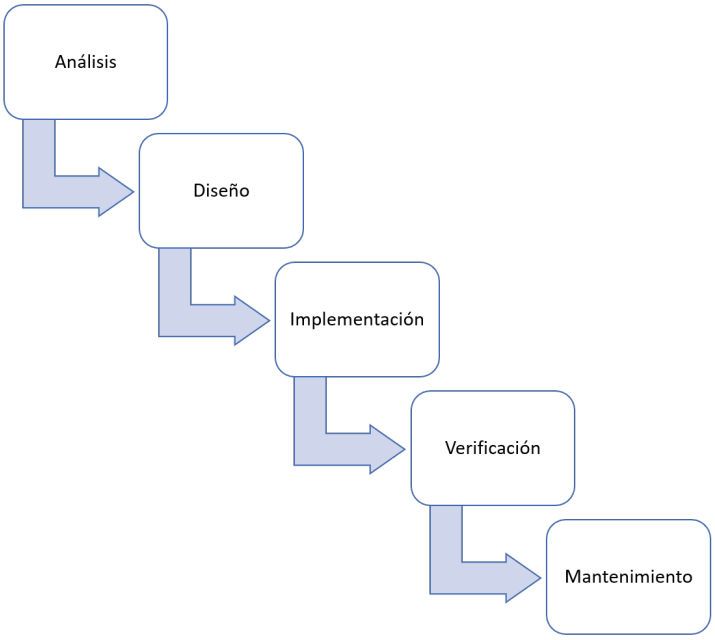

Fig. 8. Metodología de desarrollo de Software aplicada

8. Resultados

En este capítulo se presentan y analizan los resultados del proyecto realizado; para esto, se realiza la ejecución completa del software desarrollado y se mostrará el paso a paso de cada una de las dos funcionalidades que este integra. Para llegar a este capítulo se han pasado las tres últimas fases de la metodología de desarrollo. En la fase de implementación, se ha desarrollado el software en lenguaje Python 3.6 y en el sistema Operativo Ubuntu 18.04. Para la parte de verificación del sistema se han realizado diferentes pruebas, y luego de aplicar algunas modificaciones obtenidas en la fase de mantenimiento, se obtienen los resultados que se presentarán a continuación. Para efectos de una mejor presentación de los resultados, la interfaz gráfica de usuario será presentada inicialmente, seguida de la interfaz de adquisición de datos y finalmente, la interfaz de procesamiento y visualización del modelo final.

8.1. GUI Desarrollada

La interfaz principal del sistema (Fig. 9) cuenta con 4 botones, el primero permite al usuario acceder al sistema de adquisición de datos, el segundo, cuyo nombre dice “Modelo 3D”, se encarga de lanzar la interfaz para procesar los datos y generar los modelos finales. El tercer botón permite la configuración del tamaño de pantalla y el último termina la ejecución del software.

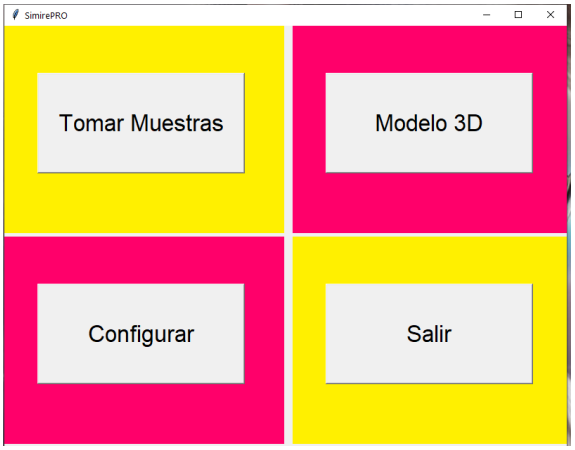

Fig. 9. Interfaz principal

8.2. GUI: Adquisición de Datos

Tal y como se había planteado en la fase de diseño, la interfaz gráfica para la adquisición de datos (Fig. 10) contiene un botón para la selección del directorio donde se desea guardar la información. Este botón abre una ventana emergente que permite el desplazamiento dentro del explorador de archivos y seleccionar la carpeta en la que deseamos se guarden las muestras del proyecto en desarrollo. Una entrada de tipo texto, permite el ingreso del nombre del proyecto y su validación para evitar sobre escribir proyectos capturados.

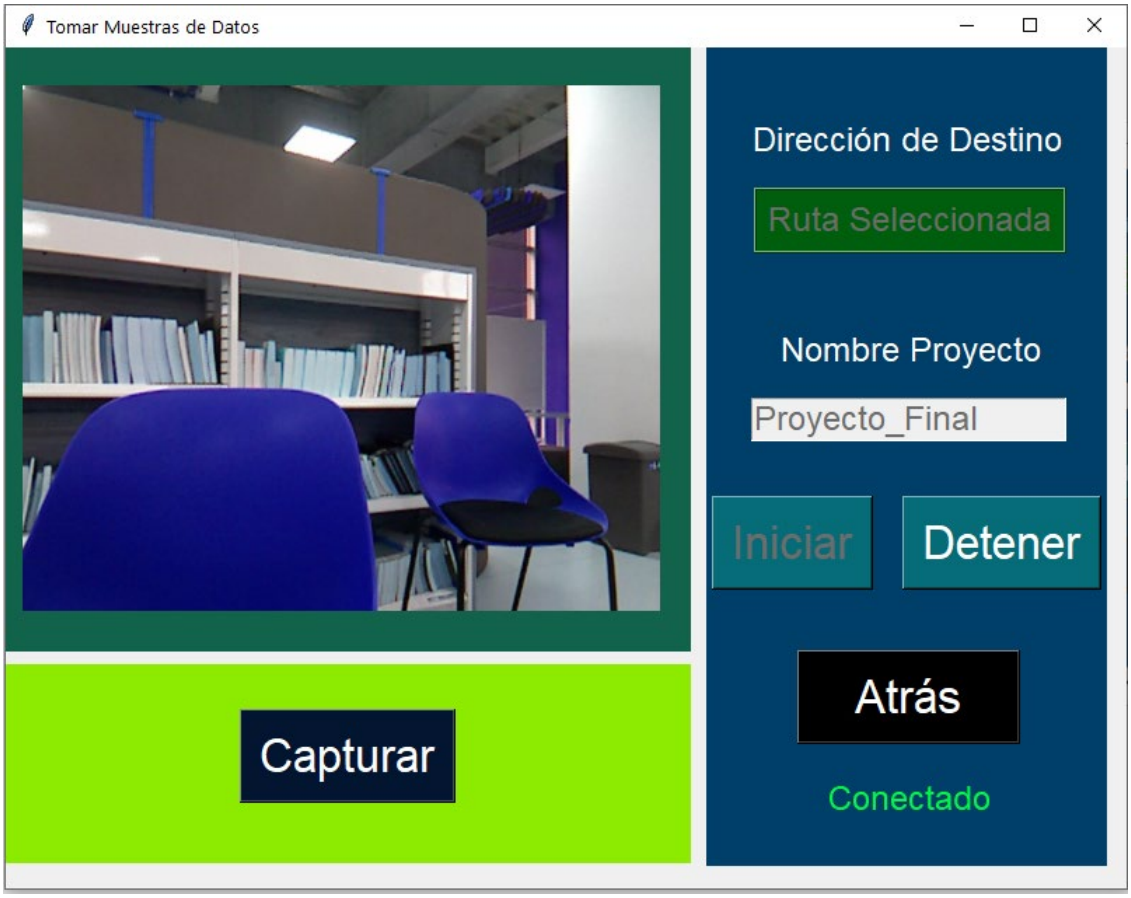

Fig. 10. Interfaz gráfica del sistema de adquisición de datos

Por otro lado, están los botones de acción, el botón de Iniciar se encarga de validar que la información ingresada sea válida para proceder a la toma de muestras. Se puede observar que, hay un apartado que nos muestra lo que el sensor está enfocando en tiempo real y se tiene a la parte inferior derecha un letrero que dice "Conectado" y hace referencia al estado del Kinect. Mientras el letrero indique un estado positivo del Kinect será posible almacenar datos, si este cambia a "Sin conexión" el software emitirá la alerta respectiva y no dejará

seguir adquiriendo datos. La Adquisición de datos se realiza de manera manual, cada vez que se oprime el botón de “Capturar”; se recomienda seguir la Fig. 6 para obtener los datos de manera fiable. Aunque en la interfaz no se muestran los datos de profundidad, el sistema se encarga de guardar tanto el mapa de colores, como el mapa de profundidad.

Para un pequeño proyecto realizado con este software, en el que se requerían 5 muestras de la escena, los datos almacenados quedan tal y como se observa en la Fig. 11; en esta podemos observar que los datos se guardan con los nombres clave (depth e img) propuestos en las fases de análisis y diseño y con los formatos (csv o archivo de valores separado por comas y jpg para los mapas de profundidad y mapas de color respectivamente). También los nombres incluyen la numeración requerida para el procesamiento de los datos.

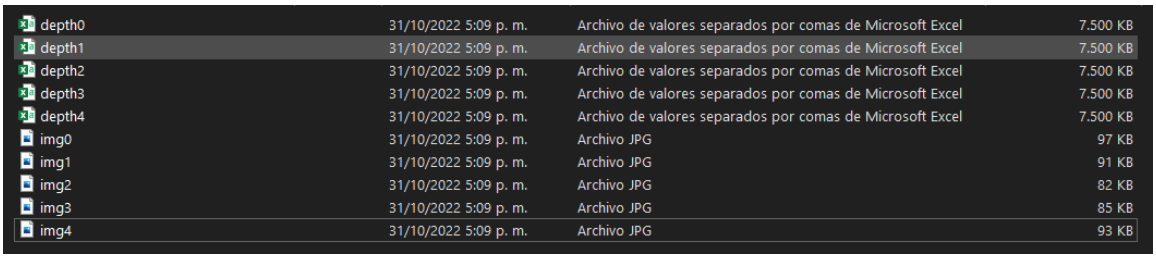

Fig. 11. Datos almacenados para un proyecto de 5 muestras

8.3. GUI: Procesamiento de los Datos

Esta interfaz tiene más elementos que la anterior, sin embargo, no todos son requeridos para la obtención de un Mosaico con Fusión de información de profundidad; dentro de los valores obligatorios se encuentran las rutas de origen de archivos y destino del archivo final, el nombre del proyecto, y la distancia entre las capturas. El filtro es opcional, en caso de no ingresar algún valor, se tomará como referencia para la distancia máxima el valor de 400 centímetros y de 50 para la mínima. La unidad de referencia tanto para el filtro de distancia como para la distancia entre capturas, es centímetros.

Fig. 12. Interfaz Gráfica de Usuario para el procesamiento de Datos y generación del Mosaico con fusión de información de profundidad

Al ingresar los datos del proyecto que se desea procesa y obtener un modelo 3D manipulable, se oprime el botón de Generar; una vez realizada esta acción, el sistema iniciará a procesar los datos con los valores ingresados. El tiempo de procesamiento dependerá mucho de las características del computador sobre el cual se está ejecutando y de la cantidad de datos que tenga que procesar, pues no es lo mismo, generar un modelo con 5 mapas de profundidad con sus respectivas imágenes, a generar un modelo a partir de 100 mapas de profundidad con sus respectivas imágenes.

Fig. 13. Mosaico con fusión de Información de Profundidad con muestras cada 20cm y sin filtro de distancia

Al finalizar el procesamiento de los datos, se abrirá una ventana (Fig. 13) en donde se visualizará el modelo final generado y permitirá su observación. Ahora, ejecutando la misma escena, pero esta vez con un filtro de distancia para tomar todos los objetos en un rango de 150cm y 300cm para quitar una de las dos sillas que se observan, y los objetos que se encuentran más al fondo de la escena (Ver Fig. 14).

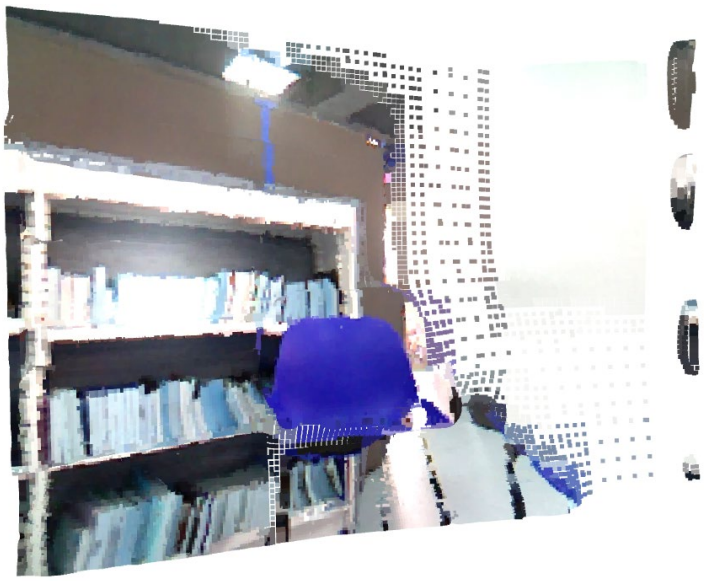

Fig. 14. Mosaico con fusión de Información de Profundidad con muestras cada 20cm y filtro de distancia de 150cm y 300cm para mínimo y máximo respectivamente.

Para la concatenación de estos datos, se ha utilizado un medidor de distancia láser de la marca Milesey que permite la medición de distancia de entre 0 y 100m con una precisión de $\pm$1.5mm. La distancia desde el Kinect hasta la silla que ha sido eliminada es de 172cm. Para el caso de los objetos del fondo, la distancia medida es de 382 cm. Sin embargo, es de destacar que el rango del Kinect es de 50cm a 400cm. Por tanto, la medida queda justo dentro del rango de medición del sistema.

9. Conclusiones

9.1. Problemas Encontrados

A lo largo de la ejecución del proyecto se han encontrado diversos problemas que con el tiempo y mediante la metodología de desarrollo de software utilizada, se han solucionado, sin embargo, existen algunos detalles que no han podido tener un buen final.

Por un lado, se encuentra el diseño de la interfaz, no ha sido posible diseñar una interfaz con características "Resposive" o de adaptación a cualquier pantalla, lo cual, hace que, en algunas pantallas, el software no se vea de la mejor manera. Otro aspecto, ya no de apariencia, sino de procesamiento de datos, y es el filtrado mediante inteligencia artificial para mejorar la calidad de la nube puntos; mejorando esto, la calidad del Mosaico con fusión de información de Profundidad tendrá mejores características.

En cuanto a software, se intentó corregir el error que tiene el Kinect por ser un sensor estéreo, sin embargo, el dispositivo en su versión 1, que es la usada en este proyecto, es demasiado inestable y muy difícil de calibrar, por tal motivo se ha desistido de esta configuración y se limita el proyecto a asumir que el dispositivo no es estéreo e ignorar el error entre la imagen RGB y el mapa de profundidad de la misma escena.

Y finalmente, un problema encontrado en la compatibilidad de las librerías de Python utilizadas en el desarrollo del software, lo que ha limitado al uso de Python en su versión 3.6 en la base del sistema operativo. Técnicamente desarrollar software sobre la base del S.O. no es recomendado, pues se podrían generar daños sobre el mismo y obligarían, generalmente, a la reinstalación de todo el sistema.

9.2. Proyectos Futuros

El proyecto desarrollado en este documento, es de propósito general, con el objetivo de que pueda ser utilizado como base para futuras investigaciones, y pueda ser aplicado en diferentes campos. Dentro de las futuras investigaciones a partir de este proyecto, se propone:

- Solucionar los problemas expuestos en la sección anterior.
- El proyecto actual se limita a la utilización del Kinect V1, por tanto, una o varias investigaciones futuras, radicaría en analizar la compatibilidad de otros

dispositivos como el LIDAR con el procesamiento de datos planteado en esta investigación.

- Permitir la manipulación del modelo final, para eliminar, de manera manual, elementos y obtener mejores acabados.
- Realizar estudios en aplicaciones concretas, dentro de las cuales puedo proponer: en la agricultura, crear análisis de crecimiento de hierba, a partir de los filtros de distancia, detección de baches a partir del filtro de distancia, reconstrucción de superficies.

9.3. Conclusiones Finales

Con el desarrollo de este proyecto se obtiene como producto final un aplicativo para el sistema operativo Ubuntu 18.04, que permite capturar datos de color y profundidad de una escena a partir del Kinect V1 para su posterior procesamiento de datos. Dentro del procesamiento de datos se analiza una estructura de escena general, y a partir de dicho análisis se genera una ecuación para generar la superposición de imágenes de un mosaico, para obtener un modelo tridimensional a partir de este y la fusión con la información de profundidad.

El desarrollo de la ecuación para la superposición de datos, se puede considerar un filtro que se encarga de eliminar los datos coincidentes entre dos fotografías, a partir de la estimación de posición desde donde se han capturado. Al eliminar información compartida entre dos conjuntos de datos, se reduce el tiempo de procesamiento final, pues el número de valores es menor, comparado con la cantidad de valores que suman las dos imágenes originales. Dado que se asume que la posición del sensor de profundidad y la cámara se encuentran en la misma posición en el dispositivo de adquisición de datos, es posible eliminar los mismos datos de la matriz de color y de la matriz de valores de profundidad.

Teniendo la matriz final, resultante de la superposición de imágenes, y de igual manera, la matriz final de valores de profundidad correlacionados al vector de color, ha sido creada una nube de puntos, ésta radica en la forma como se grafican los datos. Para graficar una nube de puntos, se debe graficar los valores de la matriz de profundidad (x, y son la posición

en el plano, y z es la profundidad de la escena), y utilizar los valores de la matriz de color, para indicar la tonalidad del punto graficado.

A partir de la nube de puntos, fue posible obtener un mesh o malla; para ello se procesaron los datos, de tal manera que, el primer paso fue crear el conjunto de vóxels representativos de la escena. El vóxel se crea a partir de la fusión de la información de profundidad con la información de color, que esencia, es lo que indica el título de este proyecto. Para crear el mesh final, se debió realizar el proceso de estimación de vectores normales de la escena, pues, con el modelo de datos inicial, estos valores no existían. Los valores normales que describen la superficie fueron requeridos, por el método de reconstrucción de superficies utilizado (método de Poisson). El formato del archivo final es ***ply*** que es compatible con diferentes plataformas de modelado tridimensional.

La interfaz gráfica desarrollada cumple con los requerimientos necesarios planteados en el análisis y diseño del software, y permiten una interacción amigable para desarrollar proyectos de generación de Mosaicos con fusión de profundidad e integra también, uno de los detalles más curiosos de esta investigación: el filtro de distancia. El filtro de distancia, se implementa con la interfaz, pero a nivel de ejecución, se aplica antes de obtener una nube de puntos final, lo que permite reducir aún más los tiempos de procesamiento de datos, para modelos con filtros seleccionados. El filtro cumple su objetivo de permitir seleccionar un rango de distancias en centímetros definido por el usuario, para seleccionar o excluir objetos de interés.

Adicional a esto, es de comentar que, la posición de los puntos de la nube generada es relativa y se toma desde el foco del sensor de profundidad hasta el objeto reconstruido. Esto sucede porque el Kinect no tiene un sistema de posicionamiento global, pues no es requerido y tampoco necesario para el enfoque del proyecto. Por otro lado, el mesh generado, conserva las medidas iniciales de la escena, aunque es difícil realizar una respectiva medición de este, esto es comprobable a partir del filtro de distancia que se aplica.

Finalmente, es importante resaltar que, tras la realización de este proyecto, se comprende la importancia del desarrollo de técnicas de visión artificial para el procesamiento rápido de datos y su implementación en herramientas que puedan llegar a hacer más fácil la vida de las personas; aunque este proyecto no se realiza para una aplicación puntual. En el apartado de

proyectos futuros, se proponen varios proyectos de aplicación que pueden llegar a cumplir tal propósito.

9.4. Estrategias de Divulgación

9.4.1. Certificados

El proyecto ha sido registrado ante la Dirección Nacional de Derecho de Autor, en donde le día 12 de octubre de 2022 se ha radicado el "CERTIFICADO DE REGISTRO DE SOPORTE LOGICO – SOFTWARE" bajo el título de **SISTEMA DE RECONSTRUCCIÓN DE MOSAICOS CON FUSIÓN DE INFORMACIÓN DE PROFUNDIDAD** (Ver Anexo 1)**.**

Referencias

[1] J. Li, W. Gao, Y. Wu, Y. Liu, and Y. Shen, "High-quality indoor scene 3D reconstruction with RGB-D cameras: A brief review," *Computational Visual Media*, vol. 8, no. 3. Tsinghua University, pp. 369–393, Sep. 01, 2022. doi: 10.1007/s41095-021-0250-8.

[2] T. Madeira, M. Oliveira, and P. Dias, "Enhancement of RGB-D image alignment using fiducial markers," *Sensors (Switzerland)*, vol. 20, no. 5, Mar. 2020, doi: 10.3390/s20051497.

[3] H. F. Murcia, S. Tilaguy, and S. Ouazaa, "Development of a low-cost system for 3d orchard mapping integrating ugv and lidar," *Plants*, vol. 10, no. 12, Dec. 2021, doi: 10.3390/plants10122804.

[4] S.-B. Ng, K.-W. Ng, R. W. O. K. Rahmat, and Y.-J. Yoong, "Blending of three-dimensional geometric model shapes," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 27, no. 1, p. 102, Jul. 2022, doi: 10.11591/ijeecs.v27.i1.pp102-109.

[5] J. Caro, "Fotogrametría y modelado 3D: un caso práctico para la difusión del patrimonio y su promoción turística," 2012.

[6] Y. Wan, J. Wang, J. Hu, T. Song, Y. Bai, and Z. Ji, "A study in 3D-reconstruction using kinect sensor," in *2012 International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM 2012*, 2012. doi: 10.1109/WiCOM.2012.6478374.

[7] V. E. L. Ang, F. E. O. Decinal, N. B. Linsangan, and J. J. Adtoon, "Volume Approximation Using Kinect Sensor," in *2021 IEEE 13th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management, HNICEM 2021*, 2021. doi: 10.1109/HNICEM54116.2021.9731913.

[8] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with Microsoft Kinect sensor: A review," *IEEE Trans Cybern*, vol. 43, no. 5, pp. 1318–1334, Oct. 2013, doi: 10.1109/TCYB.2013.2265378.

[9] R. A. El-Laithy, J. Huang, and M. Yeh, “Study on the use of Microsoft Kinect for robotics applications,” in *Record - IEEE PLANS, Position Location and Navigation Symposium*, 2012, pp. 1280–1288. doi: 10.1109/PLANS.2012.6236985.

[10] Andújar D, Ribeiro A, Bengoechea JM, San-Martín C, Fernández-Quintanilla C, and Dorado J, “RECONSTRUCCIÓN 3D DE MALAS HIERBAS UTILIZAN-DO CÁMARAS DE PROFUNDIDAD,” 2015.

[11] R. Ruoff, “Aplicación para la obtención de un mapa 3D de una escena con un dispositivo Android,” 2015.

[12] D. W. McClune, “Joining the dots: Reconstructing 3D environments and movement paths using animal-borne devices,” *Animal Biotelemetry*, vol. 6, no. 1, May 2018, doi: 10.1186/s40317-018-0150-6.

[13] P. Rodríguez, “La fotogrametría digital automatizada frente a los sistemas basados en sensores 3D activos,” *Revista de expresión gráfica arquitectónic*, pp. 100–111, 2012.

[14] L. Valhma, “3D reconstruction using Kinect v2 camera,” 2016.

[15] E. Cinar, “3D Reconstruction Technique with Kinect and Point Cloud Computing, AWERProcedia Information Technology & Computer Science,” 2013. [Online]. Available: http://www.world-education-center.org/index.php/P1754.Availablefrom:http://www.world-education-center.org/index.php/P-ITCS

[16] A. Harjoko, R. M. Hujja, and L. Awaludin, “Low-cost 3D surface reconstruction using Stereo camera for small object,” in *2017 International Conference on Signals and Systems (ICSigSys)*, 2017, pp. 285–289. doi: 10.1109/ICSIGSYS.2017.7967057.

[17] R. Siv, I. Ardiyanto, and R. Hartanto, “3D human face reconstruction using depth sensor of Kinect 2,” in *2018 International Conference on Information and Communications Technology (ICOIACT)*, 2018, pp. 355–359. doi: 10.1109/ICOIACT.2018.8350702.

[18] H. Chen, W. Chen, and T. Gao, “Ground 3D Object Reconstruction Based on Multi-View 3D Occupancy Network using Satellite Remote Sensing Image,” in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, 2021, pp. 4826–4829. doi: 10.1109/IGARSS47720.2021.9554612.

[19] S. Kamyab, H. Taghadosi, and Z. Azimifar, “A Deep Learning Embedded System for 3D Human Face Shape Reconstruction From a Single Image,” in *2022 27th International Computer Conference, Computer Society of Iran (CSICC)*, 2022, pp. 1–6. doi: 10.1109/CSICC55295.2022.9780484.

Anexos

Anexo 1: Certificado de Registro de Software

Anexo 2: Manual de Usuario

Anexo 3: Artículo