




Triangulación de puntos usando campos aleatorios de Markov con 2 imágenes consecutivas

Román Rivera Luis Rogelio , Pedraza Ortega Jesús Carlos , Aceves Fernandez Marco Antonio, Gorrostieta Hurtado Efrén, Ramos Arreguín Juan Manuel

Universidad Autónoma de Querétaro. Facultad de Ingeniería.
 caryoko@yahoo.com

Resumen

El objetivo de este trabajo es obtener la triangulación de puntos en una nube 3D para ser usada en un proceso SLAM (Simultaneous Localization And Mapping) usando un método directo el cuál utiliza la información a nivel pixel como base en la detección de puntos de interés en imágenes consecutivas en el tiempo capturadas por una sola cámara fotográfica o de video creando al final del proceso una nube de puntos en 3 dimensiones. El diseño y arquitectura del proceso requiere seleccionar entre diferentes opciones basándose en las características deseadas del proceso SLAM como tal, este diseño se basa en [1] donde se proporciona a detalle una guía y un resumen del estado del arte y áreas de oportunidad de investigación y desarrollo para los procesos SLAM. En la etapa inicial, para obtener un cuadro clave se usan campos aleatorios de Markov para calcular una función costo generando como resultado un flujo óptico de dos imágenes consecutivas, la función costo esta basada en [2], la cuál es minimizada por medio de gradiente descendente estocástico [3]. A partir del flujo óptico se calculan las posiciones de la cámara y las proyecciones en 3D son trianguladas usando la matriz esencial [4] con los puntos ajustados mediante los parámetros intrínsecos de la cámara utilizada. Los métodos tradicionales no realizan el proceso de correspondencia de puntos como se propone en el presente trabajo.

Palabras clave: campo aleatorio de Markov, gradiente descendente, triangulación

Abstract

The goal of this work is a triangulated 3D cloud points usefull in a SLAM process (Simultaneous Localization And Mapping) via a direct method wich use pixel intensities as a base to detect interest points in consecutive frames in time captured by only one video or picture camera. The design and architecture of the process requires to select from different options based in the desired capabilities of the SLAM process, based in [1] where a detailed survey of the state of the art, a guide about the design of slam processes and its coponents, and oportunity areas are provided. In the initialization stage in order to get a key frame a Markov random field is used to compute a energy fuction wich applied in two consecutive frames generates an optical flow, the energy function is based in [2], this energy function is minimized using Stochastic Gradient Descent [3]. From the optical flow camera positions are computed and 3D projections are triangulated using the esencial matrix [4], where the adjusted points with the camera intrinsics are used. Traditional methods do not process correspondences as it is proposed in this work.

keywords: Markov random field, gradient descent, triangulation

1. Introducción

El uso de cámaras digitales se ha generalizado, se pueden encontrar cámaras en dispositivos móviles, tabletas. Aprovechar la información generada por estos dispositivos para construir un modelo en 3D de la escena puede aportar información útil para diversos escenarios como navegación, mapeo, reconocimiento de objetos, reconocimiento facial, entre otros.

En un proceso SLAM basado en cuadros claves, la asociación de datos puede ser directa, basado en características y/o en una combinación de ambos. Los métodos directos se caracterizan por explotar la información de todos los píxeles, un método semi-denso solo utiliza la información de un subconjunto de píxeles. Los métodos basados en características se enfocan en regiones con gradientes sobresalientes. Los métodos híbridos combinan los métodos descritos [1].

En la etapa de inicialización de un proceso SLAM se puede utilizar una homografía asumiendo una escena plana, la matriz esencial, asumiendo una escena no plana, ó una asignación de profundidad aleatoria requiriendo procesar cuadros adicionales a fin de converger posteriormente la profundidad. SVO [5] utiliza una homografía, DT-SLAM [6] utiliza matriz esencial ambos métodos asociando características, DSO [7] y LSD-SLAM [8] utilizan valores de profundidad aleatorios. ORB SLAM [9] utiliza homografía ó matriz esencial seleccionando la opción mas adecuada mediante la penalización de un error. Se continua la investigación en métodos robustos en la asociación de los datos (correspondencias) ante cambios de iluminación, escenas y ambientes con dinámicos así como en una inicialización robusta que no requiera asumir una escena inicial [1]. El presente trabajo propone un método para la triangulación de puntos donde la adquisición de imágenes 2D consecutivas en el tiempo utiliza una sola cámara, y el proceso para obtener correspondencias, un método semi-directo utiliza un campo aleatorio de Markov basado en [2], el cuál es minimizado por medio de gradiente descendiente estocástico. El método propuesto es de utilidad en la etapa inicial de un proceso SLAM para la triangulación de una nube de puntos inicial, [3].



Figura 1: Características detectadas en un proceso SLAM [10]



Figura 2: Resultado del proceso SLAM (nube de puntos) [10]

En la Fig. 1 se muestra las características detectadas en la asociación de datos de un



Figura 3: Resultado del proceso SLAM Denso [11]

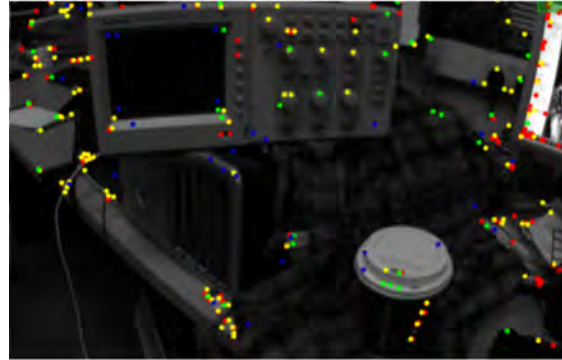


Figura 4: Características detectadas [11]

proceso SLAM, mediante las cuales se triangulan los puntos de la Fig.2 como resultado, en la Fig. 3 se muestra el resultado de la triangulación de una asociación de datos directa densa con algunas correspondencias mostradas en la Fig. 4.

1.1 Métodos y Materiales

Se propone un procedimiento, utilizando el cálculo del flujo óptico a partir de una función basada en campos aleatorios de Markov y minimizada con el método de gradiente descendiente estocástico como se muestra en la Fig. 5 en la etapa de inicialización de un proceso SLAM, etapa donde se requiere tomar una imagen como cuadro clave y generar mediante una segunda imagen, correspondencias, por medio de las cuales se calculan las posiciones de las cámaras en las primera y segunda imagen y se triangulan las correspondencias proyectando una nube de puntos 3D.

1.2 Captura de Imagen

En el presentetrabajo se realiza la captura de las imágenes como se muestra en la Fig. 5 mediante la cámara web integrada de una laptop y mediante la cámara integrada de un dispositivo móvil. Las imágenes se almacenan y procesan en una laptop Macbook.

1.3 Conversión a Grises, Preprocesamiento, 1er y 2do Cuadro

Se selecciona el 1er y 2do cuadro Figs. 6 y 7 de las imágenes capturadas, los cuadros pueden ser preprocesados, se convierten a escala de grises, opcionalmente se aplica la detección de bordes por medio del algoritmo *canny edge detection* [12] el cuál permite reducir el tiempo de computo al ciclo de enfriamiento de la función de energía.

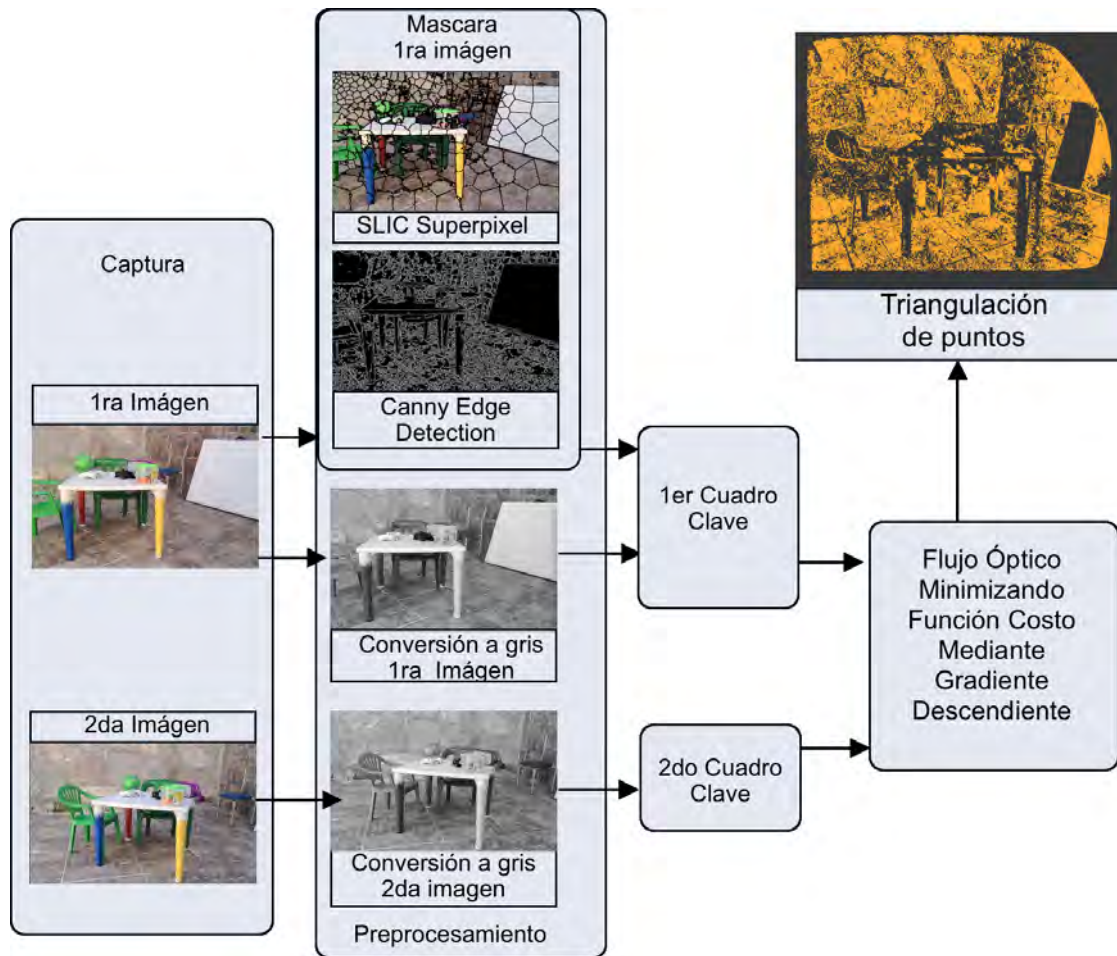


Figura 5: Método propuesto basado en la etapa inicial SLAM [1].

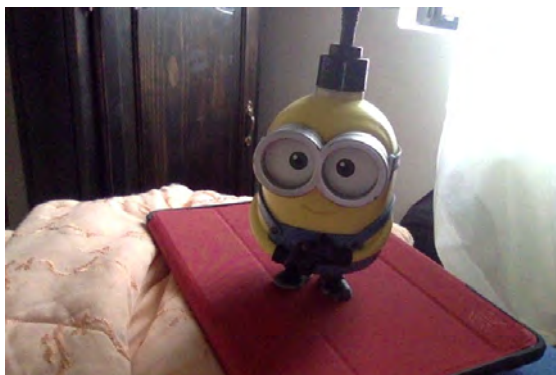


Figura 6: Figura ejemplo cuadro clave.



Figura 7: Figura ejemplo 2do. cuadro



1.4 Estimación del Flujo Óptico

El flujo óptico se refiere al movimiento aparente del campo de visión 2D entre imágenes consecutivas en el tiempo. Es diferente al movimiento de los objetos en la escena, en el caso extremo en el movimiento en el eje de la cámara, no hay flujo óptico, por el contrario rotando la cámara hay flujo óptico.

Existen 2 métodos base para estimar el flujo óptico:

- 1) Lucas - Kanade. El cuál genera un flujo de vectores disperso asumiendo un movimiento local constante y un brillo de cada pixel constante [13].
- 2) Horn Schunck. Genera un campo de flujo denso asumiendo un flujo de campo suave espacialmente [14].

Se obtiene el flujo óptico a nivel de pixel a partir de dos imágenes tomadas de una escena de manera consecutiva en el tiempo, se genera una lista de correspondencias de un conjunto de pixeles en la primera imagen a la segunda imagen. Se utilizan las Eqs. 4 y 5 para el computo del flujo óptico a nivel de pixel.

El cálculo del flujo óptico y la triangulación de las correspondencias requieren que la cámara utilizada se calibre, evitando de este tipo de distorsiones antes de procesar las imágenes. Se utilizan un proceso de calibración basado en [15].

1.5 Función Costo ó de Energía

Se utiliza una función de energía ó función costo basada en [2].

Se definen los pixeles correspondientes a una imagen de tamaño $h \times w$ donde p_{ij} corresponde al i ésimo renglón y a la j ésima columna, y el flujo óptico correspondiente a este pixel P_{ij} $F = F_{ij}$, con $i = 1, \dots, h$ y $j = 1, \dots, w$ se definen los siguientes factores en un *Markov Random Field* ó campo aleatorio de Markov:

1.5.1 Factor de Intensidad

$\phi_I : \phi_I(F_{ij}) = \exp(-E_I(F_{ij}))$ donde $E_I(F_{ij})$ es la intensidad de Energía del flujo óptico, se define para cada pixel como:

$$E_I(F_{ij}) = \frac{(I(p_{ij}) - I'(p_{ij} + F_{ij}))^2}{\alpha^2} \quad (1)$$

Donde I es la intensidad de pixel en la primer imagen e I' es la intensidad de pixel en la segunda imagen, el parámetro α es la varianza en la intensidad de la energía para controlar cuanto se compara el movimiento con la misma intensidad con movimiento con cambio en intensidad.

1.5.2 Factor de Distancia

$\phi_D : \phi_D(F_{ij}) = \exp(-E_D(F_{ij}))$ donde $E_D(F_{ij})$ es la distancia Energetica del flujo óptico, se define para cada pixel como:



$$E_D(F_{ij}) = \frac{|F_{ij}|^2}{\beta^2} \quad (2)$$

Donde $|F_{ij}|^2$ es la distancia del vector de flujo F_{ij} y el parámetro β controla la varianza de la distancia de la energía, de esta manera se favorecen vectores de flujo óptico con una energía menor, por lo tanto movimientos cortos.

1.5.3 Factor de Vecindad

$\phi_N : \phi_N(F_{ij}) = \exp(-E_N(F_{ij}, Fn_{ij}))$ donde $E_N(F_{ij}, Fn_{ij})$ es la energía en la vecindad definida en cada par de pixels adyacentes $(p1_{ij}, p2_{ij}) \in E$:

$$E_N(F_{ij}, Fn_{ij}) = \frac{|F_{ij} - Fn_{ij}|^2}{\lambda^2} \quad (3)$$

Donde $|F_{ij} - Fn_{ij}|^2$ es la distancia del vector de diferencia F_{ij} y Fn_{ij} y λ , cuando la energía de las vecindades es similares, la energía resultante es baja, permitiendo a los pixeles de un mismo objeto moverse juntos y suavizando transiciones de flujo en contornos.

1.5.4 Energía Total

La energía total del modelo probabilístico $E(\mathbf{F})$ se define como:

$$\mathbf{E}(\mathbf{F}) = \sum_{(i,j) \in \mathbf{E}} E_I(F_{ij}) + \sum_{(i,j) \in \mathbf{E}} E_D(F_{ij}) + \sum_{(i,j) \in \mathbf{E}} E_N(F_{ij}, Fn_{ij}) \quad (4)$$

$$\mathbf{E}(\mathbf{F}) = \sum_{(i,j) \in \mathbf{E}} \frac{(I(p_{ij}) - I'(p_{ij} + F_{ij}))^2}{\alpha^2} + \sum_{(i,j) \in \mathbf{E}} \frac{|F_{ij}|^2}{\beta^2} + \sum_{(i,j) \in \mathbf{E}} \frac{|F_{ij} - Fn_{ij}|^2}{\lambda^2} \quad (5)$$

1.5.5 Minimización de la función de energía.

El uso de una función de energía o función costo requiere minimizar el error resultante. En dos imágenes consecutivas, si estas son capturadas con un intervalo de tiempo pequeño considerando una velocidad constante, se asume que el flujo óptico en un pixel de manera general se encuentra localizado en una subárea alrededor de la posición de dicho pixel en la primera imagen. Se selecciona el método de Gradiente Descendiente Estocástico considerando lo anterior para minimizar la función de energía.

El método Gradiente Descendiente Estocástico para optimizar funciones costo es utilizado comunmente en redes neuronales [3].

$$\theta = \theta - \eta \nabla J(\theta; x^{(i)}; y^{(i)}) \quad (6)$$

Por cada correspondencia de los puntos de interés de la primera imagen a la segunda, se crea un ciclo de enfriamiento para la función de energía 4 donde el criterio de paro es lo que se cumpla primero, un error menor a $e \leq 0,0001$ ó la cantidad de ciclos de enfriamiento

límite de 70, en donde la función de energía converge, se encuentra una correspondencia. Los valores mostrados se eligieron por medio de una heurística.

1.6 Triangulación y cálculo de la posición de la cámara.

El proceso de Triangulación consiste en los siguientes pasos:

- Generar por medio de los parámetros intrínsecos de la cámara la matrix esencial e indexar puntos inusuales de todos los puntos mediante la función *findEssentialMat* de OpenCV.
- Generar la matriz de rotación y traslación de la camara en la segunda imagen mediante la función *recoverPose* de OpenCV
- Corregir la distorsión de los puntos que convergieron mediante la función *undistort* de OpenCV.
- Triangular las correspondencias en el espacio.

La triangulación se realiza seleccionando 4 puntos sobre una base *affine* y despues calculando los puntos restantes sobre la misma base mediante *Singular Value Decomposition* [4].

Los puntos se almacenan en formato PLY para poderlo visualizar mediante el programa Blender.

2. Experimentación

El experimento se lleva a cabo en una Macbook con un procesador de 1.1 Ghz Intel Core M3 y 8 GB de memoria RAM. El proceso se programa en C++ y se utiliza OpenCV para la captura de las imágenes y para el almacenamiento de las mismas. La triangulación se realiza con funciones y procedimientos disponibles en OpenCV.

Se toman dos imagenes consecutivas, mediante la camara web de la computadora Macbook, en este ejemplo las de la Fig. 6 y la Fig. 7 de 480x720 pixeles.



Figura 8: Cuadro clave en gris.

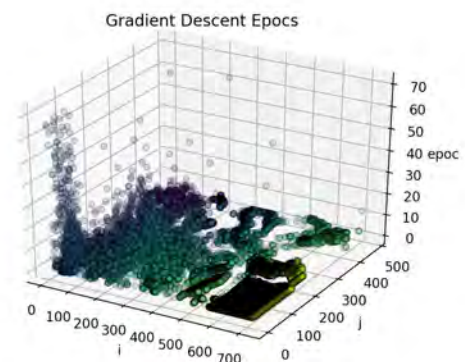


Figura 9: Ciclos de enfriamiento Gradiente Descendiente imagen completa

En la Fig. 8 se muestra una imagen en gris en la que se realiza la minimización de la función de energía. En la Fig. 7 se muestran la cantidad de ciclos de enfriamiento por cada correspondencia seleccionada que logró minimizar su temperatura.

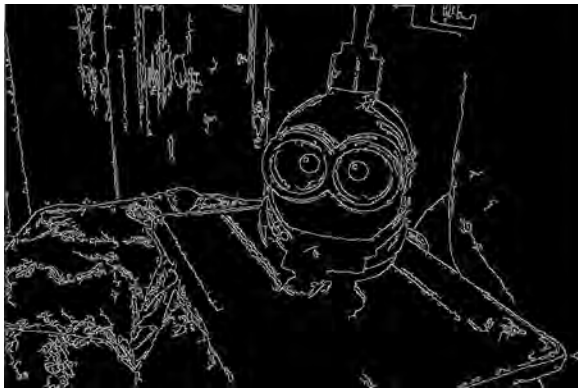


Figura 10: Cuadro Clave Preprocesado con el algoritmo *canny edge detection*.

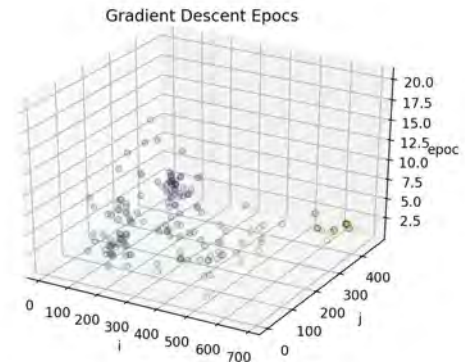


Figura 11: ciclos de enfriamiento Gradiente Descendiente imagen preprocesada con el algoritmo *canny edge detection*.

En la Fig. 10 se muestra una imagen preprocesada con el algoritmo *canny edge detection* [12]. Solo los pixeles en blanco de la Fig. 10 son entrada de la función de energía 4 y en la Fig. 11 se muestran la cantidad de ciclos de enfriamiento por cada correspondencia seleccionada que logró minimizar su temperatura.

Se encuentran las correspondencias minimizando la función de energía Eq. 5 tanto en el eje x como en el eje y para el factor de intensidad y factor de distancia en la función de energía Eqs. 1, 2 para obtener el flujo óptico de dos imágenes en donde se converge.

Después del proceso convergen 80412 en 47.9282 segundos puntos de los cuales se obtiene la siguiente triangulación de la escena:

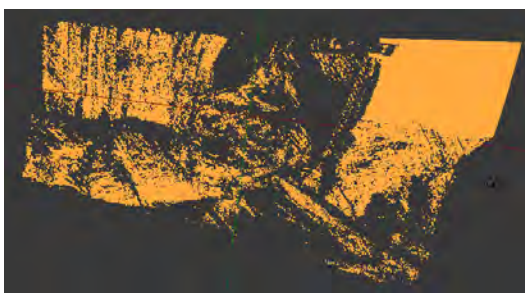


Figura 12: Nube 3D Figura 6 y 7 vista 1.

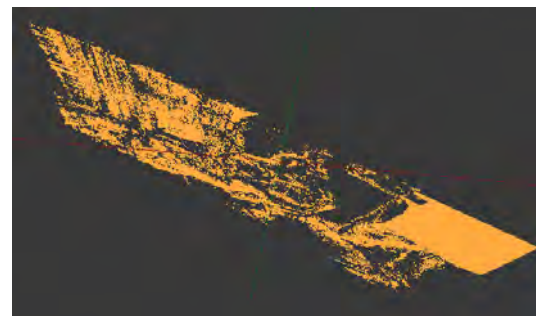


Figura 13: Nube 3D Figura 6 y 7 vista 2.

Si en el proceso de triangulación solo se toman en cuenta los puntos correctos que arroja el proceso de construcción de la matriz esencial, los puntos triangulados se reducen a 58435, el resultado es el siguiente:

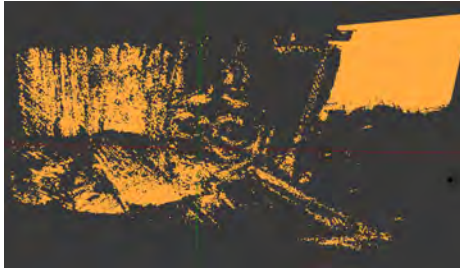


Figura 14: Nube 3D Figuras 6 y 7 puntos correctos vista 1.

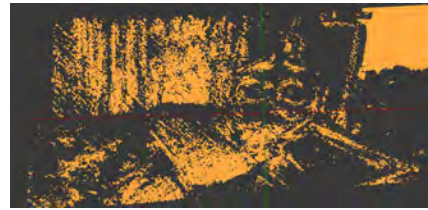


Figura 15: Nube 3D Figuras 6 y 7 puntos correctos vista 2.

Se realiza el proceso incluyendo un preprocesamiento adicional, detectando bordes con el método *canny edge detection* y solo se calculan aquellos pixeles que correspondan a la mascara generada por la detección de bordes sobre la imagen clave. El tiempo de procesamiento se reduce a 4.14785 segundos y los puntos que convergen son solamente 386.

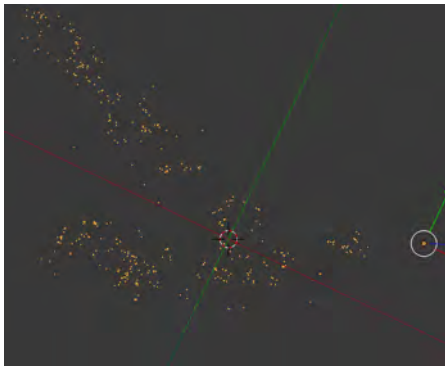


Figura 16: Nube 3D Figuras 6 y 7 aplicando *canny edge detection* vista 1.

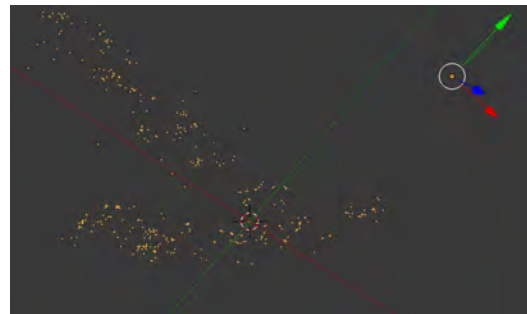


Figura 17: Nube 3D Figuras 6 y 7 *canny edge detection* vista 2.

Se toman dos imagenes consecutivas de una escena de una cocina, mediante la camara web de la computadora Macbook, en este ejemplo las de la Fig. 18 y la Fig. 19 de 480x720 pixeles.



Figura 18: Cocina 1.



Figura 19: Cocina 2.

La escena muestra objetos con distinta profundidad, existen planos como las puertas

de la cocina integral, sin embargo, son negras y no convergen o casi no convergen los pixeles que las componen como muestra en las Figs 18, 19, 20, 21

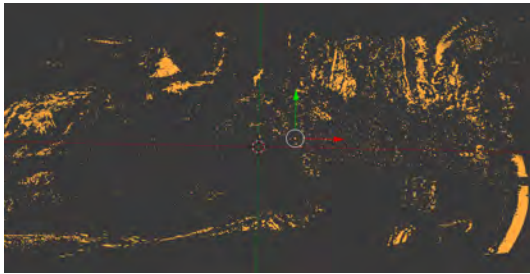


Figura 20: Cocina nube de puntos 1.

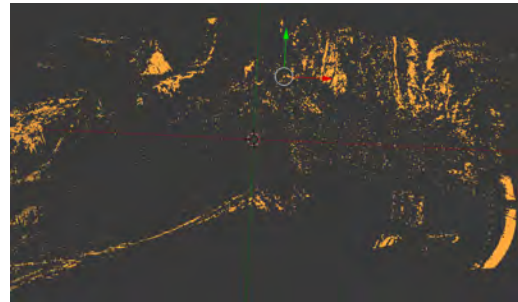


Figura 21: Cocina nube de puntos 2.

Se toman dos imágenes consecutivas de una escena de un camino, mediante la cámara de un celular, en este ejemplo las de la Fig. 18 y la Fig. 19 de 1536x2048 pixeles, se generan 1312633 puntos y el tiempo de convergencia es de 363.361 segundos como se muestra en las Figs 24 y 25.



Figura 22: Camino 1.



Figura 23: Camino 2.

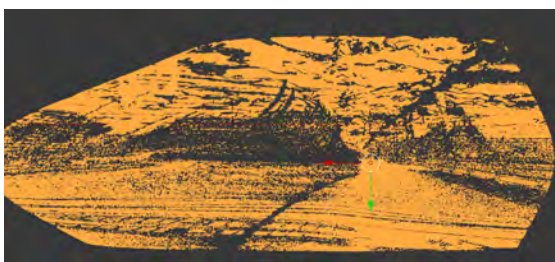


Figura 24: Nube 3D Figuras 22 y 23 vista 1.

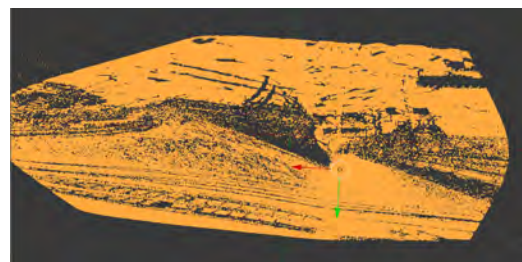


Figura 25: Nube 3D Figuras 22 y 23 vista 2.



3. Discusión y Conclusiones

La etapa de inicialización de un proceso SLAM actualmente tiene áreas de oportunidad de mejora en la robustez en escenas dinámicas y con cambios de iluminación, de igual manera hay posibilidad de mejora al buscar operar sin asumir características de una escena inicial [1].

1. La función de energía Eq. 4 basada en [2] permite obtener correspondencias entre dos imágenes consecutivas de manera aceptable bajo tolerancias a cambios de iluminación y deformaciones de objetos, observese la diferencia de iluminación en las figuras 6 y 7 de las cuales fue posible generar correspondencias y triangular una nube de puntos 3D Fig. 16.
2. El método de Gradiente Descendiente es útil para minimizar este tipo de funciones energéticas debido a la naturaleza localizada del problema donde no se requiere una solución global de minimización.
3. El procedimiento propuesto es susceptible a mejorarse por medio de etapas de preprocesamiento, la nube generada cuando se preprocesa la imagen con detección de bordes es mucho menos densa 386 puntos contra 80412 sin preprocesamiento, esta característica se puede utilizar para generar una escena en un proceso SLAM, al generar la escena e ir agregando puntos la escena puede ir creciendo en puntos acorde a necesidades variadas de densidad.
4. Se observa una mejor visualización de los puntos en escenas donde hay un pocos planos, esto es debido a que se utiliza la matriz esencial la cuál asume una escena no plana y con elementos en perspectiva en diferentes distancias.
5. La correspondencia de los puntos se puede utilizar de la misma manera en etapas posteriores de un proceso SLAM.

Referencias

- [1] Georges Younes, Daniel Asmar, Elie Shamma, and John Zelek. Keyframe-based monocular slam: design, survey, and future directions. *Robotics and Autonomous Systems*, 98:67–88, 2017.
- [2] Dongzhen Piao, Prahlad G Menon, and Ole J Mengshoel. Computing probabilistic optical flow using markov random fields. In *International Symposium Computational Modeling of Objects Represented in Images*, pages 241–247. Springer, 2014.
- [3] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.
- [4] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.
- [5] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 15–22. IEEE, 2014.



- [6] C Daniel Herrera, Kihwan Kim, Juho Kannala, Kari Pulli, and Janne Heikkilä. Dt-slam: deferred triangulation for robust slam. In *3D Vision (3DV), 2014 2nd International Conference on*, volume 1, pages 609–616. IEEE, 2014.
- [7] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2018.
- [8] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014.
- [9] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.
- [10] Ruben Gomez-Ojeda, David Zuñiga-Noël, Francisco-Angel Moreno, Davide Scaramuzza, and Javier Gonzalez-Jimenez. Pi-slam: a stereo slam system through the combination of points and line segments. *arXiv preprint arXiv:1705.09479*, 2017.
- [11] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. Dtam: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE, 2011.
- [12] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.
- [13] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *International journal of computer vision*, 56(3):221–255, 2004.
- [14] Berthold KP Horn and Brian G Schunck. Determining optical flow. *Artificial intelligence*, 17(1-3):185–203, 1981.
- [15] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22, 2000.