

# Algoritmos de navegación para robots móviles usando visión estéreo

Agustín Olmedo, *Estudiante, UBA*,  
Fernando Bugni, *Estudiante, UBA*.

**Abstract**—El presente trabajo desarrolla una implementación para la navegación de un robot utilizando visión estéreo. Al mismo utiliza conceptos sobre visión computacional: calibración estéreo, mapa de disparidad, rectificación. La implementación fue desarrollada íntegramente en C++ utilizando la librería OpenCV, además de otras librerías para la movilidad del robot.

**Index Terms**—Visión Computacional, OpenCV, Mapas de Disparidad, calibración estéreo, geometría epipolar, reproyección, región de interés, visión robótica, navegación.

## 1 INTRODUCCIÓN

LA navegación de robots es el área encargada de desarrollar robots capaces de desplazarse de forma autónoma. Esta tecnología se utiliza en distintos ámbitos como por ejemplo: misiones espaciales, misiones militares y asistencia para personas con discapacidades.

En particular en el presente trabajo desarrollaremos un algoritmo de navegación de robots utilizando visión estéreo. En la actualidad, existen diversas técnicas de visión por computadora que nos brindan la información necesaria sobre el entorno. Entre ellas se encuentra la visión estéreo que utiliza dos cámaras para obtener datos de los puntos en tres dimensiones. Algunas aplicaciones de estas técnicas son: construcción de mapas de un entorno desconocido, localización, reconstrucción 3D o también reconocimiento de objetos.

El enfoque de este trabajo consiste, principalmente, en utilizar visión estéreo para calcular el mapa de disparidad asociado a la escena capturada por las dos cámaras. Con la información de disparidad podemos obtener la distancia entre el robot y los objetos que luego será utilizada por el robot para decidir el próximo movimiento a realizar.

Un paper que ha influido fuertemente el presente trabajo fue "Region of Interest in Disparity Mapping for Navigation of Stereo Vision Autonomous Guided Vehicle". El mismo posee un enfoque similar en cuanto a las técnicas de visión utilizadas. Las diferencias radican en la interpretación de la información que nos brindan los mapas de disparidad.

La organización del paper sigue el orden cronológico en que fuimos aplicando los distintos pasos, empezando con la calibración y calibración estéreo, luego explicamos los conceptos necesarios para entender de que forma utilizamos la visión estéreo para obtener la información necesaria para realizar el algoritmo de navegación propuesto; continuando con la explicación del algoritmo de navegación. Por último, se incluye una sección de exper-

imentos donde se explica brevemente los experimentos realizado, la arquitectura del robot que se ha usado en los experimentos y las limitaciones del algoritmo.

## 2 CALIBRACIÓN DE LA CÁMARA ESTÉREO

Cuando capturamos una escena del mundo (tres dimensiones) con una cámara, ésta se proyecta sobre una imagen en dos dimensiones.

Para conocer la posición de cada punto es necesario definir un sistema de referencias, es decir, los tres ejes principales con su origen, sus orientaciones y sus unidades.

Existen tres sistemas de referencias posibles:

- El sistema del mundo
- El sistema de la cámara
- El sistema de la imagen

La unidad del sistema de referencia de la imagen es el píxel, mientras que en los demás es el metro. El sistema de la cámara tiene el origen en el centro óptico y el de la imagen en la esquina de la misma.

Los parámetros intrínsecos son los que relacionan el sistema de referencia de la cámara con el de la imagen y los extrínsecos son los que relacionan el sistema de referencia del mundo con el de la cámara (Ver figura 1).



Fig. 1. Relación entre sistemas de referencias

La conversión del sistema de referencia del mundo al de la cámara implica una traslación y una rotación.

Un punto en el mundo, visto desde la cámara, se proyecta en el plano de la imagen. Geométricamente podemos calcular la relación entre el punto en el mundo y el punto proyectado en el plano de la imagen (Ver figura 2). Además se traslada el origen del plano de la

imagen llevándolo al centro, y se realiza un cambio de unidad dividiendo por el número de píxeles por unidad de distancia.

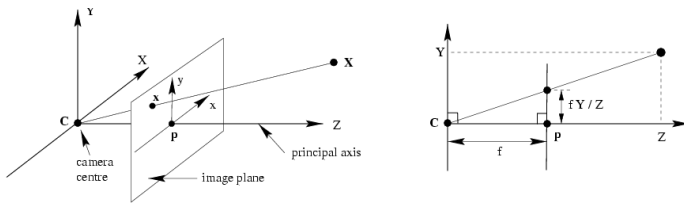


Fig. 2. *Parámetros intrínsecos: El punto en el mundo  $(X, Y, Z)^T$  visto desde la cámara se proyecta en el plano de la imagen como  $(fX/Z, fY/Z)^T$*

Con lo cual los parámetros intrínsecos son: la distancia focal, el punto principal y se agregan el skew (pixel oblicuo), y la distorsión de los lentes, etc.

La calibración de una cámara es el procedimiento por el cual obtenemos los parámetros intrínsecos y extrínsecos de la cámara a partir de imágenes tomadas por la misma.

Una vez que obtenemos los parámetros intrínsecos y extrínsecos, podemos decir que la cámara está calibrada, pudiendo relacionar cualquier punto del mundo real con los puntos proyectados en el plano de la imagen.

Para el desarrollo del presente trabajo utilizamos una cámara estéreo, con lo cual no nos alcanza solo con calibrar cada una de las cámaras, sino que además debemos calibrar las dos cámaras de forma conjunta. A esto se lo denomina calibración estéreo.

La calibración estéreo nos provee los parámetros intrínsecos de cada cámara (que son los obtenidos en la calibración de cada una) y los parámetros extrínsecos, que nos definen la posición relativa de las cámaras.

Para realizar dicha calibración utilizamos un toolbox de calibración de cámaras para Matlab <sup>1</sup>. Esta herramienta utiliza un patrón plano. El mismo debe contener cuadrados blancos y negros intercalados de un cierto tamaño conocido. El proceso del toolbox consiste en capturar imágenes de este patrón desde distintos ángulos y luego marcar las cuatro esquinas del patrón en cada una de ellas.

### 3 IMAGEN ESTÉREO

Al proyectarse los objetos, de un espacio tridimensional a una imagen bidimensional se pierde la información de la distancia a la cámara o profundidad (eje Z) de cada punto. Una forma de tratar de recuperar esta información es mediante el uso de dos cámaras, en lo que se conoce como visión estéreo.

A continuación explicaremos la configuración geométrica que se aplica en el campo de visión estéreo. La misma se conoce como geometría epipolar.

#### 3.1 Geometría epipolar

La geometría epipolar posee dos sistemas de referencias, uno por cada cámara, y surgen dos puntos importantes llamados epipolos. Un epipolo es la imagen del centro óptico de una de las cámaras con respecto a la otra (Ver figura 3).

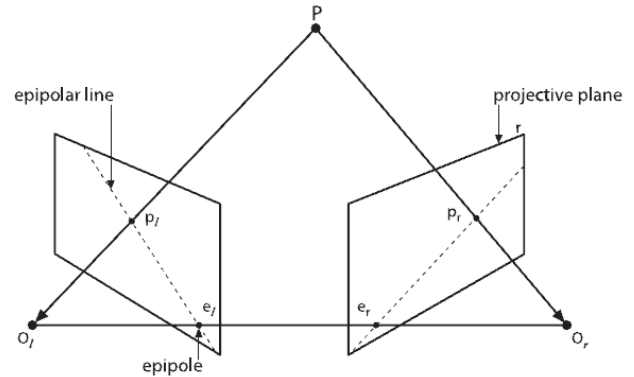


Fig. 3. Geometría epipolar

La línea que intersecta los dos centros ópticos  $O_l$  y  $O_r$  se llama línea base. Además los epipolos  $e_l$  y  $e_r$  surgen de la intersección de la línea base con el plano de la imagen de cada cámara.

Los puntos  $O_l$ ,  $O_r$  y  $P$  definen un plano llamado plano epipolar. Las proyecciones  $p_l$  y  $p_r$  así como los epipolos  $e_l$  y  $e_r$  pertenecen al plano epipolar.

Las líneas epipolares son las intersecciones del plano epipolar con los planos de las imágenes de cada cámara. Para cualquier plano epipolar, las líneas epipolares incluyen los epipolos.

La imagen de las ubicaciones posibles de un punto visto en una de las imágenes es la línea que va desde el correspondiente punto y el punto epipolar en la otra imagen. Es decir, viendo la figura 4 algún punto  $P$  en el mundo va a estar proyectado en la imagen de la cámara derecha, en la línea epipolar  $l'$ . Podemos ver como en la imagen izquierda todos los puntos se ven juntos, mientras que en la imagen de la derecha se ven alineados sobre la línea epipolar  $l'$ . Esta es la utilidad de los epipolos.

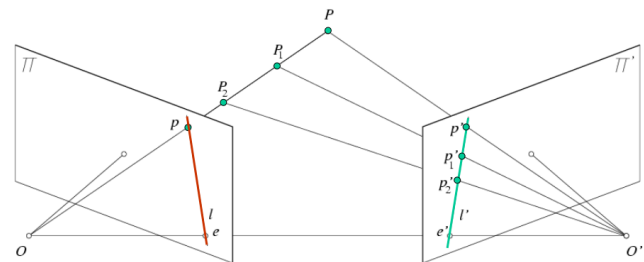


Fig. 4. Geometría epipolar

1. [http://www.vision.caltech.edu/bougetj/calib\\_doc/](http://www.vision.caltech.edu/bougetj/calib_doc/)

### 3.2 Mapas de disparidad

El algoritmo de navegación realizado en el presente trabajo utiliza información de la profundidad que existe entre los puntos del mundo y las cámaras, precisamente al centro óptico de las mismas.

Una forma de obtener la distancia a la que se sitúa físicamente un objeto en el mundo con respecto a las dos cámaras, consiste en identificar en ambas imágenes aquellos píxeles que se corresponden con la misma entidad física en la escena 3D para luego hallar la distancia que separa estos píxeles. Esto es lo que se conoce como disparidad.

Para calcular la disparidad hemos usado la librería Libelas<sup>2</sup> (Library for Efficient Large-scale Stereo Matching). Básicamente, lo que hace esta librería es buscar los descriptores característicos de cada imagen con una implementación del algoritmo SURF<sup>3</sup> (Speeded Up Robust Features) para luego buscar las correspondencias entre los puntos encontrados en el paso anterior. Con lo cual, la disparidad es  $d = x_l - x_r$ , donde  $x_l$  y  $x_r$  son las correspondencias entre los puntos de cada imagen.

Luego, conociendo el valor de disparidad y la distancia entre los centros ópticos de las cámaras (línea base) podemos calcular la profundidad de los puntos usando triángulos semejantes (Ver figura 5).

$$\frac{T - (x_l - x_r)}{Z - f} = \frac{T}{Z} \Rightarrow Z = \frac{fT}{x_l - x_r} \Rightarrow Z = \frac{fT}{d}$$

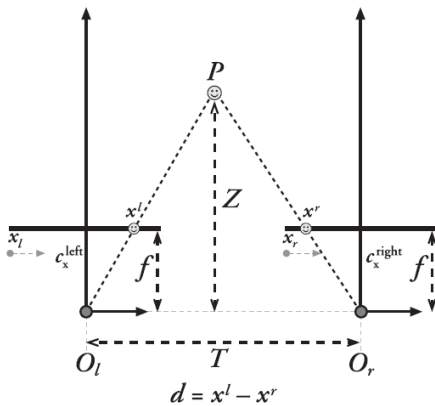


Fig. 5. Cálculo de profundidad

De la fórmula anterior podemos observar que la distancia de un punto a las cámaras es inversamente proporcional al valor de disparidad (Ver figura 6). Si el valor de disparidad es cercano a cero, pequeñas diferencias en la disparidad producen grandes diferencias en la profundidad, y si la disparidad es grande pequeñas diferencias en la disparidad no producen diferencias en la profundidad.

Con lo cual, si el valor de disparidad es muy pequeño o muy grande la disparidad no nos dice mucho sobre

la profundidad a la que se encuentra el objeto de las cámaras. En consecuencia los sistemas de visión estereó tienen buena resolución de profundidad cuando los objetos se encuentran relativamente cerca.

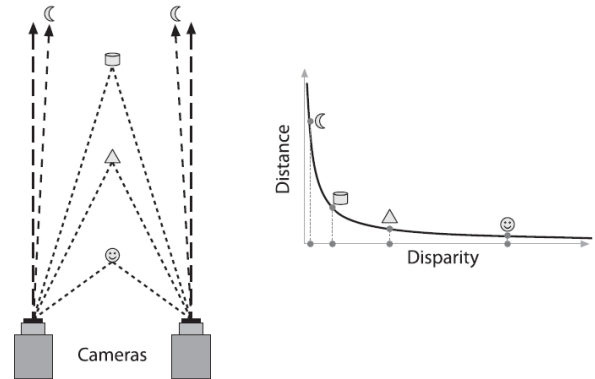


Fig. 6. Relación entre disparidad y distancia

Cabe aclarar que para conseguir el mejor resultado posible eliminaremos la distorsión radial y tangencial de las imágenes obtenidas con la cámara Minoru. Además se ajusta los ángulos y distancias entre las cámaras para que los planos de las imágenes queden coplanares<sup>4</sup>. A este último paso se lo conoce con el nombre de rectificación. Estas operaciones las realizamos con la librería OpenCV (Ver figura 7).

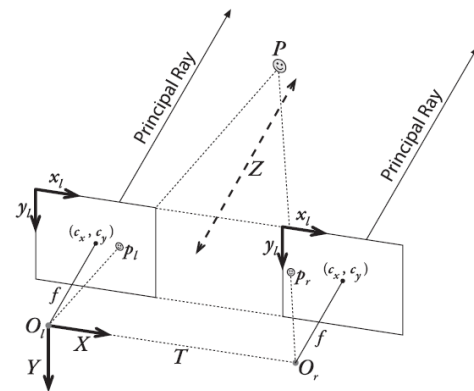


Fig. 7. Imágenes rectificadas

### 3.3 Reproyección de puntos en 2D

Conociendo los parámetros intrínsecos y extrínsecos de las dos cámaras, podemos obtener la posición absoluta en tres dimensiones de los puntos proyectados.

Como mencionamos anteriormente, para conseguir los mejores resultados posibles rectificamos las imágenes. Cuando openCV realiza la rectificación, a la vez calcula la matriz de reproyección Q.

2. <http://www.rainsoft.de/software/libelas.html>

3. [ftp://ftp.vision.ee.ethz.ch/publications/articles/eth\\_biwi\\_00517.pdf](ftp://ftp.vision.ee.ethz.ch/publications/articles/eth_biwi_00517.pdf)

4. Puntos o líneas que se encuentran en el mismo plano.

$$Q = \begin{pmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 0 & f \\ 0 & 0 & -1/T_x & (c_x - c'_x)/T_x \end{pmatrix}$$

Donde  $T_x$  es la longitud de la línea base,  $c_x$  y  $c_y$  son las coordenadas del punto principal en la primer cámara,  $f$  es la distancia focal y  $c'_x$  es la coordenada horizontal del punto principal de la segunda cámara.

Luego podemos proyectar el punto en tres dimensiones realizando la siguiente multiplicación:

$$Q \begin{pmatrix} x \\ y \\ d \\ 1 \end{pmatrix} = \begin{pmatrix} X \\ Y \\ Z \\ W \end{pmatrix}$$

Las coordenadas 3D quedarían:

$$(X/W, Y/W, Z/W)$$

## 4 NAVEGACIÓN PARA ROBOT MÓVILES UTILIZANDO LOS MAPAS DE DISPARIDAD

A continuación explicaremos brevemente los pasos realizados desde que se capturan las imágenes hasta que el robot realiza el movimiento calculado.

Como primer paso, en un mismo instante, capturamos con cada cámara la escena del mundo. Luego se calcula el mapa de disparidad, habiendo eliminado la distorsión radial y tangencial de las imágenes; así como habiendo realizado la rectificación de las mismas. Una vez que obtuvimos el mapa de disparidad se realiza una heurística para encontrar el valor medio de disparidad. Con este valor construimos el punto a reproyectar, obteniendo su posición absoluta en el mundo real. Por último calculamos el ángulo de giro y la distancia a recorrer para ir a esta posición y le indicamos al robot que se movilece a dicho lugar.

A continuación explicaremos en detalle los pasos más importantes del algoritmo.

### 4.1 Procesamiento del mapa de disparidad

Los valores de disparidad del algoritmo se asignan con el mapa de disparidad que referencia a las coordenadas de la cámara izquierda.

La región de interés es el área dentro de la cual se buscan los obstáculos. Por lo tanto, el algoritmo esta diseñando para ver los obstáculos solo en esta área particular. El tamaño de la región de interés es de 640 x 40 píxeles. Esta región esta determinada entre las filas 300 y 340 de píxeles, siendo el tamaño de las imágenes de 640 x 480 píxeles.

El procedimiento de navegación consiste en buscar un punto en la región de interés, al cual dirigirse, que este relativamente cerca del robot y en el que no se encuentre ningún obstáculo. El mismo se basa en el concepto de la figura 6. El cual nos indica que los valores de disparidad

cercanos a cero y los que son muy grandes no nos aportan información útil sobre la distancia que hay del punto a las cámaras. Para lograr esto buscaremos el valor de disparidad medio.

#### 4.1.1 Heurística del valor de disparidad medio

La heurística consiste en recorrer la región de interés calculando la mediana de los valores de disparidad de cada columna. Es decir, aplanamos el ROI<sup>5</sup> a una sola fila que contiene en cada celda el valor de disparidad medio de la columna (Ver figura 8). Luego, calculamos la mediana de esta fila consiguiendo el valor de disparidad medio del ROI.

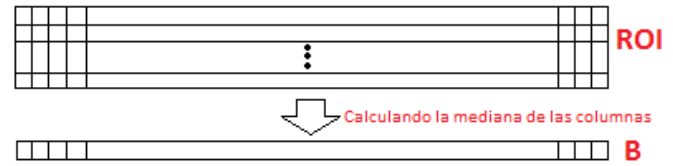


Fig. 8. Heurística

Esta heurística no nos asegura que el punto encontrado sea el mejor al cual dirigirse; pero sabemos que con dicho valor de disparidad nos brinda información fehaciente de la distancia en la que se encuentra el punto de las cámaras.

Puede ocurrir que el mapa de disparidad no nos provea suficiente información para decidir a que punto ir, ya sea porque el objeto esta demasiado cerca o porque la imagen tiene poca luz. Esto se ve reflejado en que el ROI aplanado posee más de la mitad de los valores fuera de rango. Si esto sucede se aplica una política de escape ya que la información provista no es suficiente para determinar una decisión. La política de escape consiste en girar una cierta cantidad de grados hacia la derecha buscando una escena más apropiada para encontrar un punto al cual dirigirse.

Otra ocasión en la que se aplica la política de escape es cuando el punto calculado por la heurística se encuentra muy cerca del robot. Si bien no es sumamente necesario es una mejora para encontrar mejores caminos.

### 4.2 Calculo del ángulo de giro y distancia a recorrer

Como explicamos anteriormente, para reproyectar un punto debemos multiplicar la matriz de reproyección  $Q$  por el punto a reproyectar que esta definido por:

$$(x, y, d, 1)$$

Con lo cual para reproyectar el punto al que se dirigirá el robot necesitamos definir  $x$  e  $y$ .  $x$  será la coordenada de la primer celda, del ROI aplanado, que contenga el valor de disparidad calculado por la heurística (comenzando desde la izquierda), mientras que  $y$  será la coordenada vertical inferior del ROI.

Luego re proyectamos este punto para obtener el punto P, en el mundo real, al cual dirigirse.

Por último utilizando propiedades trigonométricas calculamos el ángulo de giro y la distancia. Cabe aclarar que al robot no se le indica la distancia calculada, sino que se le indica una distancia menor para que no quede muy cerca del objeto y tenga que aplicar la política de escape.

## 5 EXPERIMENTOS

El algoritmo de navegación ha sido probado en ambientes interiores. A continuación mencionaremos algunas características que debe cumplir el ambiente para el correcto funcionamiento del mismo:

- El piso no debe tener ninguna diferencia de nivel.
- Ambientes luminosos. Si el ambiente es luminoso se consiguen mejores resultados porque la calidad de las imágenes será mejor.
- Obstáculos grandes.
- Obstáculos separados por una distancia considerable.
- Los obstáculos están apoyados en la superficie por la que se mueve el robot. Es decir, no hay objetos colgados.
- Los obstáculos no deben moverse.

### 5.1 Arquitectura de hardware

La arquitectura utilizada se puede dividir en tres partes:

- Una computadora donde se ejecuta el algoritmo descrito. La misma es una Netbook Asus con sistema operativo Ubuntu Linux, que se encuentra apoyada en un soporte que posee el robot.
- El robot, que posee una micro-computadora que corre un linux 2.6 portable. El mismo está conectado a la netbook por un cable de red (RJ-45).
- La cámara estéreo que se encuentra en la parte superior del frente del robot. La misma está conectada por USB a la netbook.

La netbook y el robot se comunican mediante paquetes UDP. Para esto utilizamos la librería libxobot-remote provista por la cátedra. La misma nos provee las interfaces necesarias para realizar los movimientos del robot, como por ejemplo, avanzar, retroceder y rotar.

En el robot se encuentra un pequeño servidor esperando los paquetes UDP y nuestro programa le envía las decisiones tomadas a este servidor que las procesa y ejecuta el movimiento.

Por otro lado, utilizamos la librería Libcam que nos permite capturar imágenes de alta calidad aprovechando al máximo la cámara estéreo.

## 6 TRABAJOS FUTUROS

A continuación mencionaremos algunas mejoras que podrían incluirse en un trabajo futuro:

- Tomar la decisión con más de una imagen. Tener un buffer de las imágenes anteriores y utilizarlas,

de alguna forma, para tomar la nueva decisión de la dirección a seguir.

- Utilizar un thread aparte para capturar las imágenes. Este thread sacaría imágenes constantemente y las dejaría en una pila y cuando el algoritmo quiera decidir un nuevo camino para continuar podría desapilar las imágenes que necesite y tomar la decisión. De esta manera podría evitarse que el robot frene entre cada pasada del algoritmo.

Además como continuación del trabajo se podría armar un mapa del lugar, visto desde arriba(2D); así como localizar el robot en dicho mapa.

## 7 CONCLUSIÓN

El mapa de disparidad es un método eficiente para almacenar la profundidad o distancia de cada píxel. Gracias a él podemos desarrollar distintas técnicas en visión robótica y utilizarlo en algoritmos que necesiten de información sobre su entorno. Un claro ejemplo se puede ver en este trabajo: utilizamos el mapa de disparidad para conducir un robot a través de una sala evitando colisiones.

A pesar de su gran utilidad el mapa de disparidad, así como también cálculo de rectificación o operaciones comunes sobre imágenes, son difíciles de programar. Requieren gran desempeño por parte del programador y poseen dificultad a la hora de probarlas. Para evitar estas dificultades debimos utilizar librerías que nos permitan manejar fácilmente las imágenes y poder sacar provecho de su información representada. *OpenCV* y *Libelas* son las librerías que nos ayudaron a superar esta dificultad. Utilizando estas librerías se puede desarrollar importantes algoritmos de visión computacional de forma rápida.

Este trabajo trata de mostrar una idea sencilla de navegación a partir del mapa de disparidad. Retoma el concepto de "Región de interés" del paper citado [1] y le da un análisis diferente para su utilización en la navegación. También su intención es poder extenderse en el futuro como describimos anteriormente.

## 8 REFERENCIAS

- [1] Anwar Hasni Abu Hasan, Rostam Affendi Hamzah, and Mohd Haffiz Johar, 2010. *Region of interest in disparity mapping for navigation of stereo vision autonomous guided vehicle*.
- [2] Gary Bradsky and Adrian Kaebler, 2008. *Learning openCV: computer vision with the openCV library*.
- [3] Richard Hartley and Andrew Zisserman, 2003. *Multiple view geometry in computer vision, second edition*.