RECONSTRUCCIÓN 3D EN TIEMPO REAL PARA UN ROBOT MÓVIL

Bustos, P.
Vicente J., Bachiller P., Broncano M.
Departamento de Informática
Universidad de Extremadura
Avenida de la Universidad s/n 10071 Cáceres
pbustos@unex.es Tfno: 927257260 Fax: 927257203
Geometría y Geometría Computacional

1. Resumen

La percepción de un entorno tridimensional en un robot móvil es aún un problema abierto para la comunidad científica. Hay dos cuestiones claves que motivan la dificultad para resolver este problema. La primera es la relación existente entre la percepción de un espacio sin objetos y la percepción de los objetos. La segunda cuestión hace referencia al concepto de objetos y a la manera de llegar a ellos. Recientemente se han realizado avances sobre la teoría geométrica de la visión estereoscópica y la visión en movimiento, que permiten iniciar nuevas líneas de exploración de estas cuestiones. La propuesta que se presenta en este artículo se basa en las teorías anteriores para generar la percepción del espacio en un robot móvil (Murphy). El sistema desarrollado es capaz de extraer información 3D visualmente y convertirla en una representación interna. Asimismo, se presentan estrategias para refinar esta representación utilizando reproyecciones sobre las imágenes y movimientos del robot.

2. Introducción

La construcción de un robot móvil capaz de percibir su entorno tridimensional es un reto que sigue resistiéndose a la comunidad científica. La razón de estas dificultades no es, en absoluto, evidente. Como es habitual existen varias líneas de trabajo, basadas en diferentes hipótesis a cerca de cómo se realiza la percepción del espacio. Desde nuestro punto de vista hay dos cuestiones claves: la primera, es la relación entre la percepción del espacio crudo, sin objetos, y la percepción de los objetos. ¿Tiene sentido esta misma distinción? y, si lo tiene, ¿es una anterior a la otra?, ¿es una distinta de la otra?. Tradicionalmente, la neurofisiología ha trabajado con la distinción entre el qué y el dónde tomando posiciones claras ante esta cuestión. Sin embargo, recientemente han surgido nuevas propuestas que sugieren una distinción entre el dónde y el cómo, reemplazando a los objetos por lo que ahora denominamos comportamientos [10]. La segunda cuestión se deriva directamente de ésta y plantea qué es un objeto o un comportamiento y cómo se llega a ellos. Desde el punto de vista de la Visión y, en especial, de la Visión en Robots Móviles la situación es tan poco alentadora como en el caso anterior. En un extremo, los objetos estaban en una base de datos 3D, en el otro, no hacen falta objetos, sólo comportamientos. El problema es que a los comportamientos les podríamos aplicar la misma pregunta sobre su origen.

Recientemente, se han realizado avances muy importantes sobre la teoría geométrica de la Visión Estereoscópica y la Visión en movimiento [6][9]. Estos resultados, junto con el incremento de la capacidad de proceso disponible, con 3DNow [8] por ejemplo, permiten iniciar líneas de exploración de estas cuestiones en condiciones de tiempo real difíciles de conseguir hasta ahora.

La postura que planteamos aquí supone una simplificación metodológica: asumimos que existe un proceso de percepción del espacio crudo, de la geometría 3D, independiente de los posibles objetos, pero no de ciertos comportamientos básicos. En concreto, la hipótesis es que el robot genera de forma activa una representación 3D del espacio que le rodea en ciclo continuo. En este ciclo, fragmentos de información 3D se integran con la representación existente y, a su vez, ésta genera acciones sobre las articulaciones del robot y sobre los elementos de proceso 2D para mantener, refinar y validar esta representación. Este planteamiento requiere una realimentación continua entre el espacio 2D y el espacio 3D, dentro de la cual se genera comportamiento físico en el robot e interno en los procesos. También, este planteamiento supone que no es posible una reconstrucción completa de la escena, a un nivel de detalle suficiente y en condiciones reales, partiendo de una única pareja de imágenes. Si bien, esto se ha conseguido en el laboratorio con proceso off-line, las técnicas empleadas distan todavía mucho de ser aplicables a un sistema de tiempo y condiciones reales, como es un robot móvil en un entorno desconocido.

En el estado actual de nuestro trabajo, el robot móvil Murphy es capaz de extraer información 3D visualmente y convertirla en una representación interna. Esta representación es, por el momento, únicamente útil para navegación o manipulación en escenas muy simples, pero los resultados obtenidos nos permiten creer que es posible llegar a una reconstrucción densa utilizando el planteamiento antes descrito.

El resto del trabajo se estructura de la siguiente forma: en primer lugar se describe el robot móvil Murphy, dotado de una torreta estereoscópica y que nos sirve como base experimental para estas investigaciones. A continuación se describe la arquitectura de proceso de todo el sistema. En el siguiente apartado se resumen las técnicas básicas utilizadas en la extracción de puntos del espacio, para concluir con una descripción del trabajo en curso sobre el refinamiento y actualización de la representación 3D.

3. El Robot Murphy

Murphy es un robot móvil dotado de seis grados de libertad que se distribuyen de la siguiente forma: dos en la base móvil con un motor en cada rueda motriz, uno en el cuello permitiéndole girar toda la torreta en bloque, uno común a las dos cámara para el movimiento de balanceo y, finalmente, uno en cada cámara para un giro independiente respecto a sus ejes verticales. Los motores son servos Futaba controlados por un microcontrolador PIC que se conecta por el puerto serie a un PC. Las cámaras son de tecnología CMOS, de bajo coste y con un gran campo de visión.

El control de alto nivel del robot se realiza desde el PC a través de un módulo software que contiene la cinemática inversa de la torreta y de la base. De esta forma es posible mandar consignas de posición en, por ejemplo, coordenadas esféricas, de tal forma que ambas cámaras converjan al punto objetivo o sigan una trayectoria 3D.

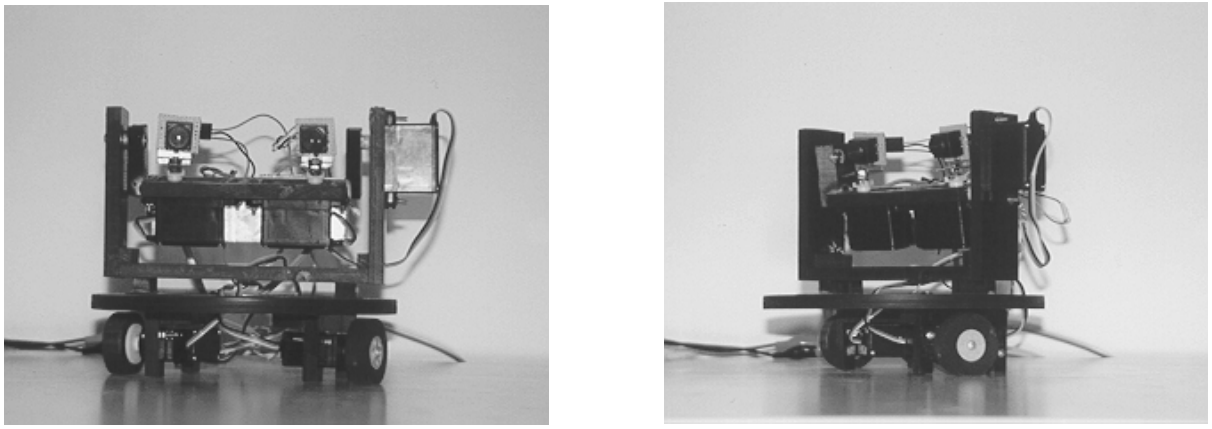

Figura 1. Vista frontal y lateral de Murphy

4. Arquitectura de proceso

La arquitectura de proceso de Murphy está estructura en tres clases, implementadas en C++. Estas clases son: Vision2D, Rep3D y Robot y se encuentran en el nivel más alto de la jerarquía. Cada una de estas clases contiene, a su vez, otras clases asociadas a subestructuras del robot y del proceso visual y motriz. La figura muestra esta estructura:

- Vision2D: recoge la estructura de la torreta estereoscópica: la traslación y rotación relativa entre las dos cámaras, como atributos de clase y, por cada cámara, una instancia de su clase. También contiene una instacia de la clase Homologos.
  - Homologos: en este objeto se realiza la búsqueda de correspondencias generando una lista de objetos asociados a parejas de esquinas, la triangulación de esquinas formando una malla y la generación de coordenadas 3D.
    - ElemHomologo: las instancias de esta clase se organizan en una lista. Cada instancia contiene dos punteros a las esquinas correspondientes y atributos donde se almacenan las coordenadas 3D.
  - Camara: contiene una instancia del driver de adquisición (v4linux2) y del detector y seguidor de esquinas Correl.
    - Correl: crea y mantiene una lista de objetos, cada uno de ellos asociado a una esquina detectada en la imagen. Para detectar las esquinas en tiempo real utiliza una librería propia de funciones 3DNow [8].
      - Elemcorr: objeto asociado a una esquina con métodos para predecir y buscar su nueva posición en la lista de esquinas suministrada por Correl.

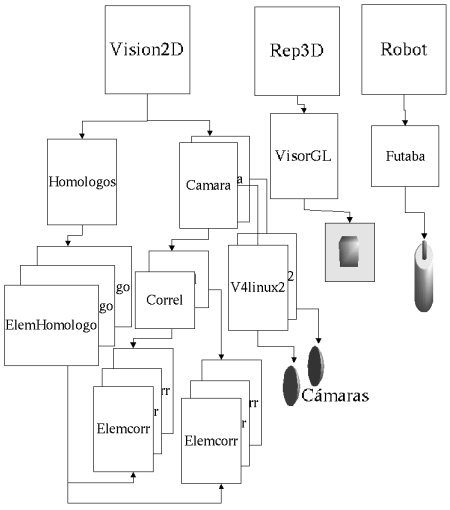

Fig. 2. Estructura de clases

- Rep3D: recibe conjuntos de triángulos 3D y construye y mantiene la representación del espacio exterior.
  - Visor: es un widget para OpenGL en el que se visualiza la lista de triángulos 3D que mantiene Rep3D. El punto de vista se ajusta a la inclinación de las cámaras respecto al suelo.
- Robot: contiene la cinemática inversa del robot y accede a la clase de control Futaba.

- Futaba : esta clase permite controlar hasta ocho servos comunicándose a través del puerto serie con la tarjeta de control.

El programa se ejecuta en un único bucle de proceso que comienza y acaba con la adquisición de las imágenes. En la implementación se ha utilizado la librería de clases Qt que permite la comunicación directa entre objetos mediante señales (signals y slots) [3].

5. Técnicas de Reconstrucción 3D

Cualquier sistema estéreo debe resolver dos problemas. El primero, conocido como correspondencia, consiste en determinar qué punto de la imagen derecha se corresponde con un punto de la imagen izquierda. El segundo problema es la reconstrucción, es decir, a partir de pares de puntos de las dos imágenes determinar la estructura 3D de la escena visualizada [7]. Para resolver estos dos problemas, el sistema propuesto lleva a cabo las siguientes fases:

1) *Detección de esquinas:* encontrar puntos de alto contraste en la imagen.
2) *Calibración de las cámaras*: determinar los parámetros intrínsecos y extrínsecos de cada cámara del sistema estéreo.
3) *Corrección de imágenes*: eliminar las distorsiones introducidas por las cámaras.
4) *Cálculo de pares de homólogos*: correspondencia entre puntos de las dos imágenes del sistema.
5) Recuperación de las coordenadas 3D de los pares homólogos.
6) *Representación 3D de la escena.*

5.1. Detección de esquinas

La búsqueda de puntos homólogos en las dos imágenes puede simplificarse considerablemente si, previamente, se extrae un conjunto de puntos cuyo entorno sea los más variado posible. De esta forma la comparación entre imágenes de estos puntos con una función de correlación obtendrá resultados mucho más precisos. A estos puntos se los suele denominar esquinas debido a que, muchas veces, coinciden con las esquinas de los objetos de la escena. Para la detección de esquinas utilizamos el filtro de Harris [7], cuyo funcionamiento se puede resumir en los siguientes pasos:

1) Suavizado de las dos imágenes mediante una convolución con una máscara gausiana de tamaño 9x9 y varianza 1.5.
2) Cálculo de la matriz de autocorrelación (2x2) del entorno 9x9 de cada pixel.
3) Obtención de los 2 autovalores de esta matriz.
4) Selección de aquellos pixels cuyo menor autovalor asociado es superior a un cierto umbral predefinido. Si hacemos que el resultado de esta umbralización sea una imagen en la que cada pixel valga 0, si no superó el umbral, o el valor del menor autovalor, si lo hizo, podemos pasar al último paso del proceso:
5) Supresión de puntos que son máximos locales mediante una búsqueda en el entorno local de cada posición. El resultado de este último paso es una imagen binarizada donde un valor distinto de cero indica la presencia de una esquina.

A continuación pasamos a la instanciación de objetos seguidores de esquinas según el esquema de clases descrito en el apartado anterior. La idea es tratar las esquinas como entidades que persisten en la secuencia de imágenes y asociar a cada una de ellas una instancia de la clase. Esto nos permite seguir la posición y velocidad de estas esquinas en el tiempo mediante un filtro de Kalman [1]. Además, esta estructura permite almacenar toda la información referente a cada esquina sin esfuerzo añadido, como las coordenadas normalizadas y rectificadas. Finalmente, y aunque no se utiliza aquí, al almacenar la trayectoria en el tiempo de una esquina, es posible asignarle un valor de fiabilidad y ordenarlas según este criterio para poder seleccionar subconjuntos más robustos en procesos de autocalibrado.

5.2. Calibración de las cámaras

Aunque existen técnicas que permiten inferir información 3D de una escena sin necesidad de conocer los parámetros de las cámaras, llevar a cabo una fase previa de calibración en un sistema de visión artificial, nos abre la posibilidad de utilizar un gran número de métodos de reconstrucción 3D y de reconocimiento. Asimismo, los modelos ópticos de los dispositivos de visualización reales obligan a tener en cuenta las distorsiones producidas por los defectos de la cámara. De esta forma, antes de realizar la reconstrucción de una escena es necesario llevar a cabo una fase de corrección de las imágenes para eliminar las distorsiones introducidas por las cámaras. Afortunadamente, dichas distorsiones pueden ser modeladas como simples distorsiones radiales, de acuerdo a las siguientes relaciones:

$$x_d = x_n(1 + k_1 r^2 + k_2 r^4)$$

Ec -1.

$$y_d = y_n(1 + k_1 r^2 + k_2 r^4)$$

Ec-2.

donde (x_n, y_n) son las coordenadas normalizadas de la proyección de un punto en el plano de la imagen, (x_d, y_d) las coordenadas del punto distorsionado y $r^2 = x_n^2 + y_n^2$. En situaciones reales, k_2 es normalmente mucho menor que k_1, por lo que el único parámetro de distorsión que debe ser estimado es k_1. Teniendo en cuenta este modelo, las coordenadas del pixel resultante de la proyección de un punto en el plano de la imagen se obtienen como:

$$[x\ y\ 1]^T = KK[x_d\ y_d\ 1]^T$$

Ec-3.

donde *KK* es la matriz formada por los parámetros intrínsecos de la cámara.

El sistema propuesto utiliza un procedimiento de calibración explícito. Es decir, realiza una estimación de los parámetros a partir de la observación de una escena en la que las coordenadas 3D de los puntos son conocidas. De este modo, el método de calibración utilizado resuelve primeramente la transformación lineal de la ecuación anterior ignorando las componentes de distorsión y, posteriormente, optimiza la estimación, incluyendo el coeficiente de distorsión radial (k_1), mediante técnicas no lineales. Para resolver este último problema el método utilizado ha sido el de Levenberg-Marquardt, ya que proporciona una convergencia rápida.

5.3. Corrección de las imágenes

Una vez estimados los parámetros de las cámaras, es necesario corregir las imágenes para eliminar las distorsiones introducidas. Para acelerar el proceso, la corrección no se realiza sobre toda la imagen, sino sólo sobre aquellos puntos que serán utilizados en la fase de reconstrucción de la escena.

Dado un determinado punto de la imagen, con coordenadas (x, y), estamos interesados en calcular las coordenadas corregidas (x_c, y_c) que vienen dadas por:

$$(x_c, y_c) = (x_n, y_n)$$

Ec-4.

donde

$$x_n = x_d / (1 + k_1 r^2)$$

Ec-5.

$$y_n = y_d / (1 + k_1 r^2)$$

Ec-6.

y

$$[x_d\ y_d\ 1]^T = KK^{-1}[x\ y\ 1]^T$$

Ec-7.

Puesto que $r^2 = x_n^2 + y_n^2$, de las ecuaciones (5) y (6) tenemos que

$$r^2 = (x_d^2 + y_d^2) / (1 + k_1 r^2)^2$$

Ec-8.

Si denotamos $x_d^2 + y_d^2$ como r_d^2, llegamos a la siguiente ecuación:

$$k_1 r^3 + r = r_d$$

Ec-9.

Por lo tanto, el cálculo de las coordenadas corregidas (x_c, y_c) se reduce a resolver el polinomio de grado 3 de la ecuación (9).

5.4. Cálculo de puntos homólogos

El rendimiento de cualquier método de correspondencia puede verse empeorado por la existencia de puntos que sólo son visibles desde una de las cámaras y por la asociación de falsos pares de puntos debido al ruido o a la falta de iluminación. Los efectos de estos fenómenos pueden reducirse si se tiene en cuenta la geometría epipolar del sistema estéreo para realizar la búsqueda de pares de homólogos.

Dado un par estéreo de cámaras, cualquier punto P del espacio 3D define un plano a través de los centros de proyección de las dos cámaras. Dicho plano (plano epipolar) corta los dos planos de imagen en dos líneas epipolares. De este modo, si para cada punto p_I de la imagen izquierda se determina la correspondiente línea epipolar en la imagen derecha, se puede restringir la búsqueda de los homólogos de p_I a aquellos puntos que estén en su línea epipolar. El problema de la correspondencia se simplifica aún más si rectificamos cada imagen de forma que los pares conjugados de líneas epipolares sean colineales y paralelas al eje horizontal de la imagen. Así, el conjunto de posibles homólogos de un punto (x_I, y_I) de la imagen izquierda estará formado por aquellos puntos de la imagen derecha cuya coordenada y sea igual a y_I.

La rectificación se lleva a cabo mediante una rotación de las cámaras alrededor de los centros ópticos que consigue transformar las líneas epipolares en rectas paralelas al vector de traslación T del sistema. Una posible formación de dicha matriz de rectificación (R_{rect}) es la siguiente:

$$R_{rect} = \begin{pmatrix} e_1^T \\ e_2^T \\ e_3^T \end{pmatrix}$$

Ec-10.

donde e_1, e_2 y e_3 son vectores definidos de la siguiente forma:

$$e_1 = \frac{T}{\|T\|}$$

Ec-11.

$$e_2 = \frac{\begin{bmatrix} -T_y & T_x & 0 \end{bmatrix}^{\mathrm{T}}}{\sqrt{T_x^2 + T_y^2}}$$

Ec-12.

$$e_3 = e_1 \times e_2$$

Ec-13.

Esta matriz de rotación permite rectificar un determinado punto de la imagen izquierda con coordenadas corregidas (x_c, y_c, 1) de la siguiente forma:

$$[x'\, y'\, z']^{\mathrm{T}} = R_{rect}[x_c\, y_c\, 1]^{\mathrm{T}}$$

Ec-14.

A partir de la ecuación anterior, las coordenadas rectificadas del punto (x_r, y_r, 1) vienen dadas por:

$$(x_r, y_r, 1) = (x', y', z') / z'$$

Ec-15.

Para los puntos de la imagen derecha hay que construir la matriz de rectificación (Rd_{rect}) teniendo en cuenta la matriz de rotación R del sistema. De esta forma, Rd_{rect} se expresaría como:

$$Rd_{rect} = R_{rect}R^{\mathrm{T}}$$

Ec-16.

La rectificación de estos puntos se lleva a cabo de la misma forma que para los puntos de la imagen izquierda (ecuaciones 14 y 15) utilizando la matriz de rotación Rd_{rect}.

Una vez rectificados los puntos de las dos imágenes, se crean los conjuntos de posibles homólogos para cada punto p_I de la imagen izquierda. Así, para un punto de la imagen izquierda con coordenadas rectificadas (x_{Ir}, y_{Ir}), su conjunto de homólogos H_I estará formado por aquellos puntos de la imagen derecha (x_{Dr}, y_{Dr}) que verifiquen $y_{Dr} \approx y_{Ir}$. Si el conjunto de homólogos H_I no es vacío, el punto homólogo de p_I será aquel que maximice el criterio de similitud para una ventana de la imagen definida alrededor del punto. La similitud entre p_I y cada punto p_D de H_I se mide como la correlación entre los puntos de imagen contenidos en las ventanas de ambos puntos:

$$c(p_I, p_D) = \frac{\sum_{i=-W}^{W}\sum_{j=-W}^{W}(I_I(x_I+i, y_I+j) - \bar{I}_I)(I_D(x_D+i, y_D+j) - \bar{I}_D)}{\sqrt{\left(\sum_{i=-W}^{W}\sum_{j=-W}^{W}(I_I(x_I+i, y_I+j) - \bar{I}_I)^2\right)\left(\sum_{i=-W}^{W}\sum_{j=-W}^{W}(I_D(x_D+i, y_D+j) - \bar{I}_D)^2\right)}}$$

Ec-17.

donde I_I e I_D son las imágenes izquierda y derecha, respectivamente, $\bar{I}_I$ e $\bar{I}_D$ los valores medios de los pixels contenidos en las dos ventanas, (x_I, y_I) las coordenadas originales de p_I, (x_D, y_D) las coordenadas originales de p_D y $2W+1$ el ancho en pixels de la ventana de correlación.

Para regiones de puntos similares, el coeficiente de correlación de la ecuación (17) dará un valor cercano a 1, por lo que se considera que p_D es homólogo de p_I si maximiza dicho coeficiente en relación con los restantes puntos de H_I, pero además verifica $c(p_I, p_D) \approx 1$.

Podemos resumir el procedimiento de cálculo de pares de homólogos en los siguiente pasos:

1) *Rectificación de puntos*: transformación de las coordenadas corregidas (x_c, y_c) de un punto de imagen a coordenadas rectificadas (x_r, y_r)
2) *Calcular el conjunto de posibles homólogos para cada punto de la imagen izquierda*: dado un punto p_I con coordenadas rectificadas (x_{Ir}, y_{Ir}), construir el conjunto de homólogos H_I de dicho punto con los puntos p_D de la imagen derecha que verifiquen $y_{Dr} \approx y_{Ir}$.
    1. *Determinar los pares de puntos homólogos*:
    - Para cada punto p_I de la imagen izquierda, obtener el punto $p_D \in H_I$ que maximiza $c(p_I, p_D)$.
    - Insertar en la lista de homólogos el par p_I-p_D si se verifica $c(p_I, p_D) \approx 1$.

5.5. Reconstrucción 3D

Dado que los parámetros intrínsecos y extrínsecos del sistema estéreo son conocidos, es posible llevar a cabo la reconstrucción 3D de la escena utilizando un método de triangulación. Fijando el eje de referencia del sistema en el centro de proyección de la cámara izquierda, podemos considerar para cada par de puntos homólogos p_I-p_D, las rectas definidas por O_I-p_I y O_D-p_D, donde O_I y O_D son los centros de proyección de las cámaras izquierda y derecha, respectivamente. La intersección de ambas rectas proporciona las coordenadas 3D de un punto P de la escena. Puesto que los parámetros del sistema sólo son conocidos de manera aproximada, dicha intersección no es exacta y hay que calcular P como el punto más cercano a las dos rectas [2][7].

Podemos definir las rectas O_I-p_I y O_D-p_D de la siguiente forma:

$$O_I\text{-}p_I \Rightarrow ap_I$$

Ec-18.

$$O_D\text{-}p_D \Rightarrow bR^T p_D + T$$

Ec-19.

donde R y T son la matriz de rotación y el vector de traslación del sistema y p_I y p_D están expresados en coordenadas corregidas ($p_I = (x_{Ic}, y_{Ic}, 1)$; $p_D = (x_{Dc}, y_{Dc}, 1)$). Si definimos un vector w (ecuación 20) perpendicular a las dos rectas, el punto más cercano a O_I-p_I y a O_D-p_D es el punto medio de un segmento paralelo a w que une ambas rectas.

$$w = ap_I \times R^T p_D$$

Ec-20.

En definitiva, el problema se reduce a resolver el siguiente sistema lineal de ecuaciones:

$$ap_I + cw = bR^T p_D + T$$

Ec-21.

Este sistema proporciona los puntos de unión del segmento con las dos rectas, que vienen dados por ap_I y $bR^T p_D + T$. Así, el punto P puede determinarse como:

$$P = ap_I + cw/2$$

Ec-22.

El resultado de aplicar este proceso a cada par de homólogos obtenidos de la fase anterior es un conjunto de puntos 3D, que puede ser utilizado para inferir las características de la escena visualizada y llevar a cabo las acciones correspondientes.

5.6. Triangulación

Un conjunto de esquinas en las dos imágenes no es suficiente para que la reconstrucción del espacio sea útil al robot. Por otro lado, obtener una reconstrucción densa partiendo de puntos requiere un postproceso de las imágenes extremadamente costoso, por ejemplo utilizando Programación Dinámica. La alternativa que estamos explorando se basa en hacer un conjunto de hipótesis sobre la escena que nos proporcionen una estructura completa del espacio 3D. Con este punto de partida, podemos utilizar el movimiento del robot y la reproyección 3D a 2D para refinar esta representación.

La técnica utilizada consiste en hacer una triangulación de Dealunay sobre las esquinas de una de las cámaras que han sido emparajedas con sus homólogas de la otra cámara. Para ello utilizamos el software *triangle.c* creado por J.R. Shewchuk que es altamente eficiente [4][5]. El resultado de este proceso es una malla de triángulos 2D que cubren la imagen. La suposición que se hace es sobre la existencia real de los lados de los triángulos como contornos visibles de la escena. Muchos de ellos no lo serán y, por tanto, la representación 3D diferirá de la escena real. Lo que resta es convertir estos triángulos a 3D y comenzar a refinar la representación obtenida. Para la conversión no hay que hacer nada más, puesto que los vértices son el conjunto de homólogos y, para ellos ya se han calculado las coordenadas 3D. El ajuste y validación de la representación se discute a continuación.

6. Representación del espacio

Murphy representa la escena visualizada como una malla de triángulos en el espacio, esto es, como una superficie continua aproximada por planos. Esta representación tiene varias ventajas frente a otras técnicas como NURBS, para el esquema que planteamos. Por ejemplo, es inmediato calcular homografías planares, reproyectar puntos o regiones en las cámaras y asignarles una textura directamente desde la imagen. Además, su visualización es muy eficiente y es fácil de refinar, añadiendo o eliminando triángulos.

Aunque, de momento, el sistema se limita a crear y mantener esta representación, el objetivo inmediato es que este módulo genere movimientos en el robot y reproyecciones sobre las imágenes para validar y ajustar la representación. La idea es partir de la hipótesis de que cada plano representado existe en el mundo y utilizar acciones que validen o refuten esta hipótesis. Las opciones que estamos explorando se basan en comparar las texturas que se llevan desde las imágenes a los triángulos, con la reproyección de estos sobre las mismas imágenes. Por el lado del movimiento, la validación consiste en hacer converger las cámaras hacia puntos significativos de la reconstrucción 3D y comprobar lo que aparece en los centros de las imágenes, después del movimiento.

Cada rectificación de la existencia o tamaño de los triángulos se incorpora a la representación como un refinamiento sobre la anterior, imponiendo las restricciones necesarias. La principal dificultad de este proceso es que el refinamiento tiene que coexistir con los movimientos de traslación y orientación del robot, así como, con la posibilidad de que haya regiones de la escena con movimiento propio. La necesidad de que todas estas fuentes de cambios se integren en una representación consistente, da lugar a lo que denominamos representación activa. Frente a una versión pasiva de este concepto, las representaciones activas generan acciones en el robot y predicen internamente sus resultados en términos de percepción. Las acciones se seleccionan para mantener la representación dentro de un compromiso entre estabilidad e innovación. Por ejemplo, si la velocidad del robot y el efecto que ésta tiene en la percepción de la escena hacen que no sea posible mantener una representación al nivel de detalle necesario para un desplazamiento seguro, las acciones posibles son: disminuir la velocidad o bien, restringir el área de la imagen a una zona central.

7. Experimentos

En la imagen siguiente se muestra una captura de pantalla del software desarrollado. En el centro de la figura puede verse una ventana con las dos imágenes capturadas por las cámaras. Sobre ellas, las cruces muestran la posición de las esquinas detectadas.

A la izquierda aparece el panel de control de los motores del robot. En la parte derecha, se muestra la tringulación de las esquinas detectadas en la imagen correspondiente a la cámara izquierda. El panel de la parte superior derecha muestra la velocidad de captura -incluida la representación gráfica. Finalmente, en la esquina superior izquierda aparece la representación 3D de los triángulos detectados.

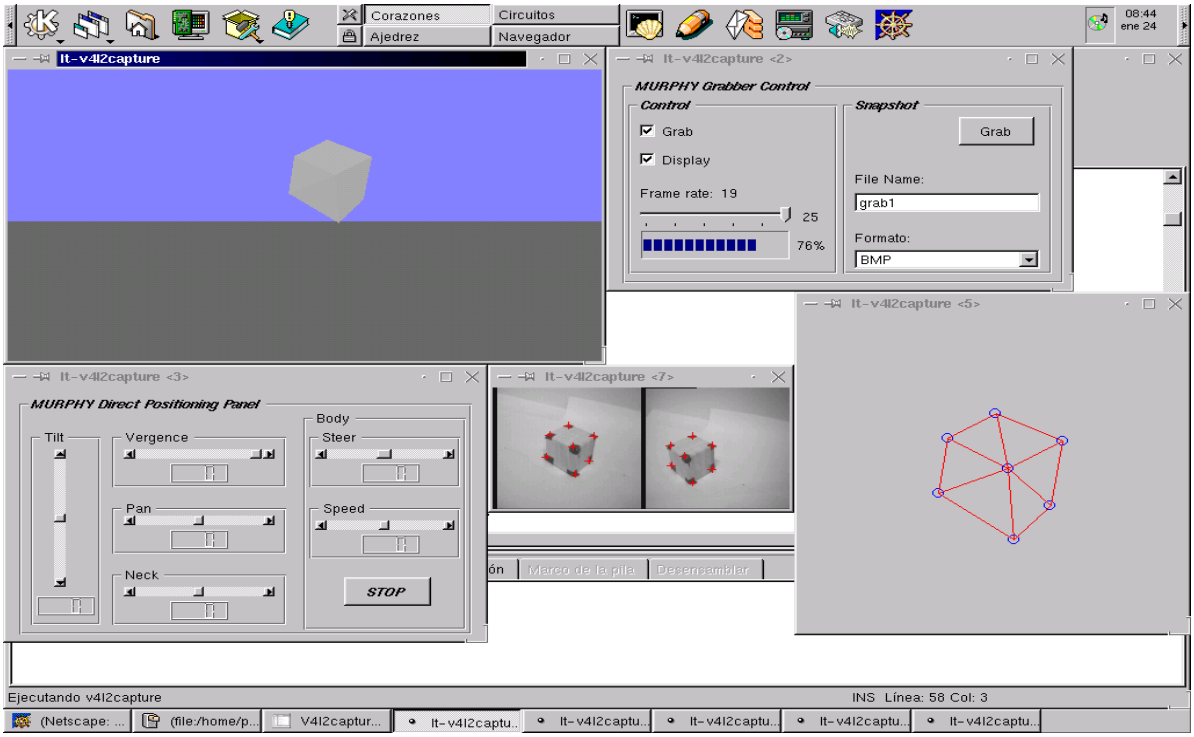

Figura 3: Entorno de trabajo del robot

8. Conclusiones

Se ha presentado un robot móvil como base de un proyecto de investigación en curso, cuyo objetivo es crear y mantener una representación 3D del espacio. Esta representación debe permitir al robot navegar y manipular de forma robusta y fiable. Los resultados presentados son los primeros pasos en esta dirección y muestran la posibilidad del enfoque planteado.

En resumen, las características del robot Murphy son las siguientes:

- Plataforma móvil con 6 gdl.
- Torreta estereoscópica.
- Procesamiento remoto en tiempo real utilizando una librería propia para 3DNow.
- Software para calibrado no lineal offline.
- Software orientado a objetos sobre Linux usando la libreria Qt.
- Utilización de la geometría epipolar para la obtención de puntos homólogos.
- Triangulación Delaunay de las esquinas 2D utilizando el software triangle.c de J.R. Shewchuk
- Obtención de las coordenadas 3D de los triángulos.
- Representación planar 3D del espacio y visualización con OpenGL.

9. Referencias Bibliográficas

[1] Grewal M. S., Andrews A. P., *Kalman Filtering. Theory and Practice*, Pretince-Hall, 1993.

[2] Hartley, R. and Sturm P., *Triangulation*. Proceedings ARPA IUW pp. 957-966, 1994

[3] Qt Library http://www.trolltech.com

[4] Shewchuk J.R. *Engineering a 2D Quality Mesh Generator and Delaunay Triangulator*. First Workshop on Applied Computational Geometry (Philadelphia, Pennsylvania), pp. 124-133, Association for Computing Machinery, May 1996. http://www.cs.cmu.edu/~quake-papers/triangle.ps

[5] Shewchuk J.R. *Software triangle.c* http://www.cs.cmu.edu/~quake/triangle.html

[6] Torr P. H. S. Beardsley, P. A. Murray, D. W., *Robust Vision*, British Machine Vision Conference, 1994, pp. 145,155, 1994

[7] Trucco E., Verri A. *Introductory Techniques for 3D Computer Vision*. Prentice-Hall, 1998

[8] Vicente J. *A 3DNow Library for Image Processing*. TR-RL 1/01 2001

[9] Zissermann A., Hartley R.. *Multiview Geometry.* Cambridge University Press. 2000

[10] Zeki S. *A Vision of the Brain*, Blackwell Scientific 1993