INSTITUTO POTOSINO DE INVESTIGACIÓN CIENTÍFICA Y TECNOLÓGICA, A.C.

POSGRADO EN CONTROL Y SISTEMAS DINÁMICOS

Estimación de pose y velocidades de objetos en movimiento mediante visión por computadora

Tesis que presenta

Pablo Báez Miranda

Para obtener el grado de

Maestro en Control y Sistemas Dinámicos

Director de la Tesis:

Dr. David Antonio Lizárraga Navarro

San Luis Potosí, S.L.P., Julio de 2014

Constancia de aprobación de la tesis

La tesis **Estimación de pose y velocidades de objetos en movimiento mediante visión por computadora** presentada para obtener el Grado de de Maestro(a) en Control y Sistemas Dinámicos fue elaborada por **Pablo Báez Miranda** y aprobada el **31 de 07 de 2014** por los suscritos, designados por el Colegio de Profesores de la División de (nombre de la división) del Instituto Potosino de Investigación Científica y Tecnológica, A.C.

_____

Dr. Arturo Zavala Río

(Presidente)

_____

Dr. David Antonio Lizárraga Navarro

(Secretario)

_____

Dr. Hugo Cabrera Ibarra

(Sinodal)

_____

Dr. Marcial Bonilla Marín

(Sinodal)

Créditos Institucionales

Esta tesis fue elaborada en la División de Matemáticas Aplicadas del Instituto Potosino de Investigación Científica y Tecnológica, A.C., bajo la dirección del Dr. David Antonio Lizárraga Navarro.

Durante la realización del trabajo el autor recibió una beca académica del Consejo Nacional de Ciencia y Tecnología CONACYT-424238 y del Instituto Potosino de Investigación Científica y Tecnológica, A. C.

IPICYT

Instituto Potosino de Investigación Científica y Tecnológica, A.C.

Acta de Examen de Grado

El Secretario Académico del Instituto Potosino de Investigación Científica y Tecnológica, A.C., certifica que en el Acta 009 del Libro Primero de Actas de Exámenes de Grado del Programa de Maestría en Control y Sistemas Dinámicos está asentado lo siguiente:

En la ciudad de San Luis Potosí a los 31 días del mes de julio del año 2014, se reunió a las 11:00 horas en las instalaciones del Instituto Potosino de Investigación Científica y Tecnológica, A.C., el Jurado integrado por:

Dr. Arturo Zavala Río	**Presidente**	**IPICYT**
Dr. David Antonio Lizárraga Navarro	**Secretario**	**IPICYT**
Dr. Hugo Cabrera Ibarra	**Sinodal**	**IPICYT**

a fin de efectuar el examen, que para obtener el Grado de:

MAESTRO EN CONTROL Y SISTEMAS DINÁMICOS

sustentó el C.

Pablo Báez Miranda

sobre la Tesis intitulada:

Estimación de pose y velocidades de objetos en movimiento mediante visión por computadora

que se desarrolló bajo la dirección de

Dr. David Antonio Lizárraga Navarro

El Jurado, después de deliberar, determinó

APROBARLO

Dándose por terminado el acto a las 12:10 horas, procediendo a la firma del Acta los integrantes del Jurado. Dando fe el Secretario Académico del Instituto.

A petición del interesado y para los fines que al mismo convengan, se extiende el presente documento en la ciudad de San Luis Potosí, S.L.P., México, a los 31 días del mes de julio de 2014.

Dr. Marcial Bonilla Marín
Secretario Académico

Mtra. Ivonne Lizette Cuevas Vélez
Jefa del Departamento del Posgrado

INSTITUTO POTOSINO DE INVESTIGACIÓN CIENTÍFICA Y TECNOLÓGICA, A.C.
IPICYT
SECRETARIA ACADEMICA

A mi familia.

Agradecimientos

La presente Tesis es un esfuerzo en el cual participaron, directa o indirectamente, varias personas leyendo, opinando, corrigiendo, aconsejando, dando ánimo o acompañándome en los momentos más difíciles así como en los momentos de felicidad.

Agradezco al Dr. David Antonio Lizárraga por haber confiado en mí para la realización de esta Tesis, por la paciencia, por aminorar mis defectos y enaltecer mis virtudes.

Hago extensivo mi agradecimiento a mis sinodales, el Dr. Arturo Zavala, el Dr. Hugo Cabrera y el Dr. Marcial Bonilla, por sus consejos y recomendaciones, tanto para el trabajo de Tesis como para la vida.

Quiero agradecer a mis padres por el apoyo que me han brindado en todo lo que he hecho. También agradezco a mi hermano Víctor por su ánimo y alegría. Gracias por su comprensión, ánimo y apoyo incondicional que me han dado.

También quiero dar un agradecimiento especial a mis compañeros y amigos que me acompañaron a lo largo de este camino que fue la maestría. Les agradezco por aconsejarme, soportarme, convivir conmigo y permitirme ser parte de sus vidas a través de estos casi tres años. Gracias Omar, Luis, Ernesto, Adriana, Roberto, Juan, Aidé, Erick, Eduardo, Aletse, Nelly, Ángeles, Héctor y Bahía.

Gracias a todos.

Índice general

Índice de figuras

Resumen

Muchos procesos automáticos requieren de la interacción de robots con su medio ambiente, para los cuales la visión computacional es una gran herramienta para el reconocimiento y localización de objetos.

En este trabajo se plantea un escenario en el que un brazo manipulador requiere de la interacción con objetos que se desplazan sobre un transportador lineal y se utiliza la visión computacional para calcular la pose y velocidad de dichos objetos. Esto utilizando una sola cámara para extraer esa información, estableciendo un marco de referencia en la cámara y un marco en las diferentes posiciones observadas del objeto.

Para lograr dicha tarea, se utilizan propiedades geométricas para extraer la relación entre dos escenas del mismo objeto, y después esta información es utilizada para estimar la pose del objeto con respecto al marco de referencia de la cámara. Del mismo modo, se muestra la utilización de un observador que estima el error de posición (que es la diferencia en la pose del objeto con respecto a una pose inicial de referencia), que al multiplicar dicha estimación por una matriz Jacobiana afín, se puede identificar la velocidad traslacional y rotacional del objeto.

Abstract

Many automatic processes require the interaction of robots with their environment, for which the computer cision is a great tool for recognition and localization of objects.

This work describes a scenario, in which a manipulator requires the interaction with objects moving on a linear conveyor, and computer vision is used to estimate the pose and velocity of those objects. All of this using a single camera to extract that information, by setting reference frame on the camera and another frame in each position observed of the object.

To accomplish this task, geometric properties are used to extract the relationship between two scenes of the same object, and then this information is used to estimate the pose of the object relative to the reference frame of the camera. Similarly, an observer that estimates the position error system (which is the difference in the pose of the object relative to an initial reference pose) is shown, and multiplying the observer estimation by a Jacobian-like matrix, one can identify the translational and rotational velocity of the object.

Capítulo 1

Introducción

El sentido de la vista nos proporciona un alto porcentaje de la información que recibimos de nuestro ambiente, lo cual nos permite realizar un sinfín de tareas. Lo anterior brinda a la Visión Artificial (o visión por computadora) un gran campo de aplicación en la robótica, en particular cuando es necesario que los robots interactúen con objetos del medio ambiente.

La función principal, en procesos industriales, de la visión computacional es localizar y reconocer estos objetos por medio del procesamiento de imágenes. Las imágenes captadas por cámaras o sensores están formadas por valores discretos de intensidad de luz y de color, que están relacionados con la iluminación, propiedades geométricas y de la superficie del objeto, la óptica de la cámara y las caracteristicas del sensor. El procesamiento de estas imágenes tienen en cuenta la identificación de puntos característicos del objeto, remoción de defectos, mejorar propiedades de brillo y color, etc. Todo esto con la finalidad de obtener de la imagen la información necesaria para nuestro uso.

La visión computacional es un campo joven que avanza rápidamente y de manera constante aparecen resultados nuevos con técnicas de visión más eficientes y que resuelven problemas cada vez más complejos y exigentes.

1.1. Motivación

La visión computacional tiene una gran área de aplicación en procesos industriales. En esta investigación de tesis se utilizan técnicas de visión computacional para calcular las posiciones y trayectorias de objetos para la aplicación robótica de una celda de manufactura con objetos en movimiento. En la situación que se plantea, la celda cuenta con un brazo manipulador que ejecuta tareas que implican la interacción con estos objetos que se desplazan en un transportador lineal. Para que dicho brazo pueda interactuar y realizar su tarea, se debe conocer la posición y orientación de estos objetos en tiempo real, lo cual puede lograrse mediante un sistema de visión por computadora.

Muchas técnicas de estimación de pose y velocidad usan cámaras digitales a manera de "sensor". Los estimados se usan como entrada en una gran cantidad de problemas de control. Esto ha motivado en el área de visión computacional un gran interés en la estimación de pose y velocidad. Estos métodos de estimación varían en la cantidad de información de entrada que requieren, en su complejidad computacional y su precisión, entre otras cosas.

En muchas aplicaciones de ingeniería, uno se ve motivado a usar una cámara para determinar la velocidad de un objeto en movimiento. Sin embargo, como se muestra en [1], el uso de una cámara requiere de la interpretación de movimiento tridimensional a través de imágenes 2D provistas por una cámara. El problema principal es que la información 3D está comprimida o transformada no linealmente en información 2D; y por lo tanto se requiere desarrollar técnicas o métodos para obtener la información 3D cuando sólo se conoce la información 2D [2].

Para abordar la detección de la velocidad del objeto, muchos investigadores han desarrollado varias aproximaciones. Por ejemplo, si se conoce un modelo del movimiento del objeto, se puede usar un observador para estimar la velocidad del objeto [3]. En [4], se utiliza un observador para estimar la velocidad del objeto, sin embargo es necesario conocer la cinemática del objeto. En [5], examinan con gran detalle el problema de identificar los parámetros de

movimiento y la forma de un objeto utilizando las ecuaciones de Riccati. En [6], utilizan un modelo en tiempo discreto usado para predecir la ubicación de puntos característicos del objeto. En [7], utilizan técnicas de predicción y filtrado de trayectorias para rastrear un objeto.

El método que se usa en este trabajo de tesis tiene como requerimiento el uso de una sola cámara.

1.2. Objetivo

El objetivo principal de esta tesis es el estudio y simulación de métodos existentes para la detección de objetos en movimiento mediante una sola cámara, con la finalidad de extraer la pose y velocidad de dichos objetos.

1.3. Historia de la visión computacional

En los últimos años ha habido grandes avances en el campo de la visión computacional. Se ha logrado desarrollar técnicas matemáticas para recuperar información de una imagen, como la forma o bordes de un objeto, reconstruir superficies tridimensionales, contrarrestar defectos de la imagen, incluso el reconocimiento de caras. Sin embargo, el problema de la visión computacional no deja de ser difícil, pues es un problema en el que se trata de recuperar información a partir de información incompleta o parcial para especificar la solución por completo.

1.3.1. Visión computacional en los 70's

En los años 70's, cuando se iniciaba el estudio de la visión computacional, se trataba de imitar la capacidad humana y dotar al robot de inteligencia artificial. En lugares como el MIT o la universidad de Stanford creían que resolver el problema de visión sería sencillo y que luego podrían pasar rápidamente a problemas de razonamiento y planeación más complejos.

Sin embargo, otros trabajos vieron la necesidad de tomar la visión computacional como una rama de estudio independiente, en el que se busca recuperar estructuras tridimensionales a partir de imágenes. Se desarrollaron técnicas para obtener los bordes e inferir la estructura tridimensional del objeto observado. Una forma común de resolver el problema era usando "cilindros generalizados", sólidos en revolución y un barrido de curvas cerradas acomodadas en partes relacionadas [8] [9]. Estos son los algoritmos más usados actualmente en el área de reconocimiento de objetos. Se desarrollaron enfoques cuantitativos de visión computacional, incluyendo algoritmos que presentaban el uso de cámaras estéreo y algoritmos de flujo óptico. También se desarrollaron enfoques cualitativos para el entendimiento de las variaciones en sombreado e intensidades de luz, y cómo estos efectos resultan en la formación de la imagen, como la orientación de las superficies y de las sombras. En 1982 David Marr resumió la gran cantidad de ideas que se tenían sobre cómo funcionaba la visión humana [10]. Marr introdujo la idea de tres niveles de descripción de un sistema de procesamiento de información visual, y que, más o menos, se resumen en:

- Teoría computacional: ¿Cuál es el objetivo computacional y qué restricciones conocidas puede tener el problema?
- Representación y algoritmos: ¿Cómo son las entradas, las salidas e información intermedia representada y qué algoritmos calculan el resultado deseado?
- Implementación de hardware: ¿Cómo los algoritmos y representaciones son mapeados a hardware real?, ¿Cómo las restricciones de hardware se pueden usar para la selección del algoritmo adecuado?

1.3.2. Visión computacional en los 80's

En la década de 1980, hubo mayores estudios en la creación de algoritmos matemáticos más sofisticados para realizar el análisis cuantitativo de imágenes y escenas. Una técnica que se volvió ampliamente usada fue el uso de pirámides para mezclar imágenes y para la búsqueda de correspondencia se desarrolló el concepto de procesamiento de *scale-space*[11],[12], [13].

1.3.3. Vision computacional en los 90's

Muchos de los trabajos de décadas anteriores siguieron en la década de 1990. Se desarrollaron técnicas de factorización para resolver el problema del uso de una cámara ortogonal que después se extendió a casos donde afecta la perspectiva. Cuando éstas técnicas se generalizaron como un problema de optimización global notaron que era lo mismo que las técnicas de *bundle-adjustment* usadas en fotogrametría. Con estas mismas técnicas se desarrollaron sistemas automáticos de modelado tridimensional. También se desarrollaron algoritmos de visión estereo de múltiples vistas para producir superficies completas en tres dimensiones que aún se siguen usando. Se siguió con el mejoramiento de algoritmos de seguimiento, incluyendo el seguimiento de contornos usando *contornos activos* tales como *serpientes* [14], *filtro de partículas* [15], así como técnicas basadas en intensidad. Estos algoritmos son normalmente aplicados para el rastreo de caras [16], [17] y cuerpos completos [18].
Otros trabajos que se realizaron en este periodo y que consistían en técnicas de aprendizaje estadístico resultaron en los primeros algoritmos de reconocimiento de caras y en sistemas dinámicos lineales para el seguimiento de curvas [19].

1.3.4. Vision computacional en los 2000's

Las técnicas de visión computacional fueron ampliamente profundizadas en la década de los 2000's, apoyadas por el uso de éstas en cámaras digitales comerciales. Otra gran tendencia que surgió en los 2000's fue el uso de técnicas de aprendizaje que reconocen rasgos que despues se usan para el reconocimiento de los objetos [20].

Capítulo 2

Marco Teórico

En este capítulo se establecerán las bases para los problemas tanto de estimación de pose como de estimación de velocidad de objetos en movimiento. Se presenta un breve resumen de algunos métodos para resolver el problema, también se presenta a qué nos referimos con pose y cómo se representa la velocidad de un objeto a partir de la expresión para su pose.

Recordemos que el escenario que se plantea en el trabajo de tesis es una celda de manufactura que se encuentra ejemplificada de manera muy sencilla en la figura 2.1. En la celda se encuentra un brazo manipulador que ejecuta tareas que implican la interacción con objetos en movimiento. Cada uno de estos objetos tiene forma constante y se desplaza sobre un transportador lineal. Es necesario que el robot conozca la posición y orientación de estos objetos, así como sus velocidades, para que realice sus tareas sobre dichos objetos.

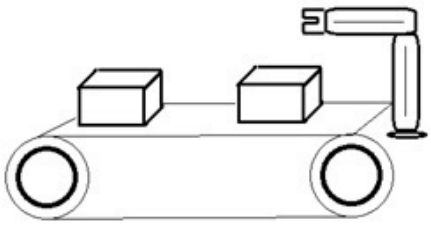

Figura 2.1: Ejemplo de celda de manufactura

El uso de una cámara como sensor requiere la interpretación de movimiento de un ob-

jeto tridimensional a través de las imágenes bidimensionales obtenidas por la cámara. El problema principal es que la información 3D se proyecta y colapsa a información 2D. La proyección es una transformación no lineal [2], y es esta información 2D a la que se tiene acceso para la estimación de la pose y la velocidad.

2.1. Estado del Arte

El problema de estimación se puede resolver de varias maneras, entre los más comunes está el uso de los filtros de Kalman, o el uso de análisis fotogramétricos. A continuación se mencionarán algunos métodos para solucionarlo.

En [21] se ataca el problema comparando dos conjuntos de puntos característicos, donde cada conjunto consiste en las observaciones de puntos del objeto de una posición de referencia y el objeto en su nueva posición. Estos puntos pueden ser en 2 o 3 dimensiones, dependiendo de cómo sean observados; es decir, si los datos fueron tomados por una cámara o un sensor de movimiento. En la misma referencia se resuelve el problema con un método iterativo de mínimos cuadrados ponderados para hacer más robusto el método.

Para el problema de estimar la pose tridimensional a partir de imágenes de video en 2 dimensiones, [22] presenta un método que usa mínimos cuadrados que minimiza la distancia de las líneas proyectadas en 2 dimensiones hacia una superficie en 3 dimensiones.

Otro algoritmo usa una aproximación descrita en [23] a través del uso de un filtro de Kalman, para la estimación de la posición tridimensional y de la orientación de un objeto en movimiento con respecto al marco de referencia establecido en un robot. El algoritmo tiene como característica el hecho de que se toma como desconocida la posición de la cámara. El algoritmo da solución al problema después de una larga secuencia de imágenes, asignando a las imágenes más lejanas menor peso o importancia, pues suelen tener más ruido.

Tambien en [24] utilizan un filtro de Kalman para resolver las ecuaciones de colinealidad fotogramétricas para obtener parámetros que resultan en la estimación de pose.

Otro ejemplo del uso de un filtro de Kalman para resolver el problema de estimación se encuentra en [25]. En esta referencia se utiliza el filtro para calcular una solución recursiva de las ecuaciones de proyección de imágenes estéreo. El algoritmo que usan se puede aplicar con cualquier número de cámaras en el espacio de trabajo siempre y cuando éstas observen ciertos puntos característicos del objeto. El algoritmo proporciona una estimación robusta, incluso con la existencia de ruido, distorsión o errores de calibración.

Por otro lado, [26] muestra un esquema iterativo para obtener la orientación relativa, que no requiere de una suposición inicial para obtener la pose de objeto. El método requiere de un conjunto de pares de proyecciones correspondientes a puntos centrales de la escena vistos de dos cámaras diferentes.

En [27] se muestra un método para calcular la posición y orientación de un objeto conocido a partir de 4 puntos característicos coplanares. El algoritmo que presentan empieza con una aproximación ortográfica, e iterativamente se aproxima a un estimado de la pose.

Para el problema de estimación de velocidad, en [28] utilizan un método de dominio mixto espacio-temporal, simplemente conocido como método mixto. El método mixto es conocido por su robustez ante ruido y cambio de forma de un objeto en movimiento. Pero el método mixto requiere de la extracción del fondo, pues éste afecta la estimación. En el método que muestran en [28] solucionan el problema de extracción del fondo aplicando al método mixto a una imagen de un punto característico del objeto.

Algunos métodos requieren el uso más de una cámara. Por ejemplo, en [29] muestran un sistema de visión estereoscópica que estima la velocidad y dirección de un objeto en movimiento. Muestran cómo, a través de operaciones vectoriales y de geometría tridimensional, son capaces de extraer la magnitud del vector velocidad del objeto obesrvado, así co-

mo su dirección.

Para resolver el problema, nosotros usaremos el algoritmo descrito en [30], el cual integra el algoritmo de estimación de pose propuesto en [31] y el estimador de velocidad de [2]. El algoritmo requiere del uso de una sola cámara fija. Además requiere que los objetos observados cuenten con una cara plana y que se puedan extraer 4 puntos característicos de esta cara a través de algún algoritmo de procesamiento de imágenes (en la figura 2.2 se muestra un ejemplo sencillo). El algoritmo primero obtiene la pose del objeto a través de una reconstrucción geométrica, la que posteriormente funciona como entrada para el estimador de velocidad.

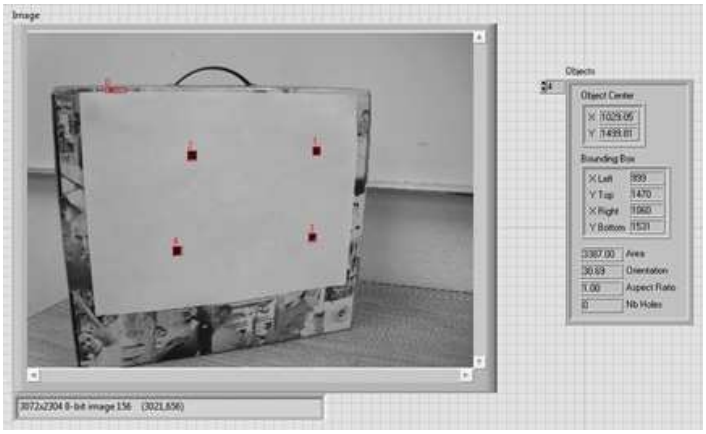

Figura 2.2: Ejemplo de interfaz de un programa de identificación de puntos característicos en LabView

2.2. Definición de pose y velocidad

Para representar la pose de un objeto de manera matemática, usamos un método usado comúnmente en mecánica que se muestra en [32]. Considerando la figura 2.3, se adjunta un marco de referencia inercial A a la posición de origen y sus 3 ejes coordenados se denotan con i_0, j_0 y k_0. También se adjunta un marco de referencia, denominado B, en el objeto y sus ejes coordenados i_1, j_1 y k_1. Siendo A el marco de referencia en el órigen, y B el marco de referencia acoplado al objeto.

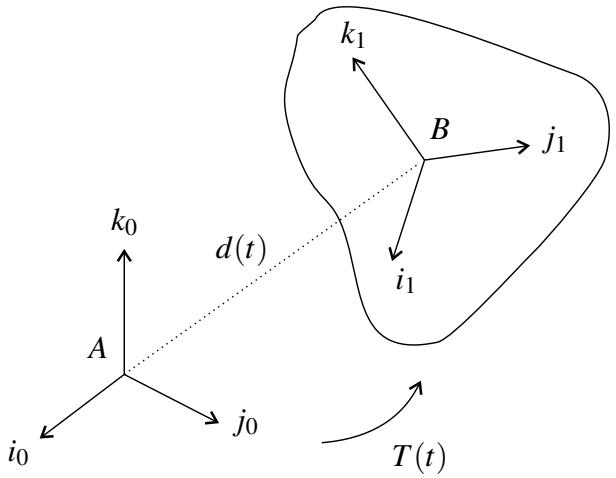

Figura 2.3: Pose de un objeto

La posición del objeto es expresada con las coordenadas de su origen con respecto de A, y la orientación del objeto viene determinada por la matriz de rotación cuyas columnas son las componentes de i_1, j_1, k_1, respectivamente, en términos de la base $\{i_0, j_0, k_0\}$ Es decir, los ejes del marco de coordenadas

$$B = \{i_1, j_1, k_1\}$$

componen la matriz

$$R = (i_1 j_1 k_1)$$

Donde $R \in SO(3)$, el grupo especial ortogonal, que representa la parte rotacional del cambio de posición del marco B con respecto a A, es una matriz de rotación. Así, podemos expresar la pose como

$$T_m = \begin{pmatrix} R & t_r \\ 0 & 1 \end{pmatrix} \tag{2.1}$$

Donde R es la rotación antes mencionada y $t_r \in \mathbb{R}^3$ representa la traslación dada por la posición del origen de B con respecto a A.

Para expresar la velocidad del objeto podemos, al igual que se hace para la pose, expresarla en términos de la velocidad, tanto rotacional como traslacional, del marco B fijo al objeto, con respecto al marco de referencia fijo A. La velocidad traslacional del marco en el

objeto se puede expresar como la derivada en tiempo del vector de posición P, que representa el origen de B con respecto a A.

$$v_t = \frac{dP(t)}{dt} \tag{2.2}$$

Pero para expresar detalladamente la expresion de velocidad rotacional del marco de referencia hay varias maneras. Una forma es elegir un vector ϕ, cuyas componentes sean 3 coordenadas que expresen la orientación en función de t y calcular su derivada con respecto al tiempo $\dot{\phi} = d\phi/dt$. Las 3 variables que componen ϕ pueden ser diferentes dependiendo de la representación que se esté usando, puede ser en ángulos Euler, o ángulos "roll, pitch y yaw", etc.

Otra manera es usando un tercer marco de coordenadas, un marco auxiliar B' que tiene el mismo origen que el marco A, y que tiene la misma orientación que el marco B. Es decir, ambos marcos de referencia, tanto B como B', rotan de la misma manera. Entonces, la rotación de B' con respecto de A en cada instante de tiempo, se puede expresar en términos de un eje que pasa por el origen. Estosignifica que la velocidad de B', y por lo tanto la velocidad rotacional de B, puede ser descrita por un vector w; este vector es llamado *vector de velocidad angular*.

Cuando se usa el primer método para describir la velocidad,la pose se pude expresar con un vector T_v definido por

$$T_v = \begin{bmatrix} P \\ \\ \phi \end{bmatrix} \tag{2.3}$$

donde P y ϕ son vectores de 3 componentes que expresan la posición y la orientación respectivamente, es decir, la pose de B con respecto a A. Y la velocidad del marco del objeto B con respecto al marco de referencia A esta dada por la ecuación

$$\dot{T} = J_T(d)\dot{d}$$

Donde d es el vector de desplazamiento del objeto y $J_T(d)$ es la matriz jacobiana de T con respecto a d, está dada por

$$J_T(d) = \frac{\partial T}{\partial d^T}.$$

Por otro lado, en el segundo método, el cual se usará usando en este trabajo para expresar la velocidad rotacional, el vector

$$v = \begin{bmatrix} \dot{P} \\ w \end{bmatrix} \tag{2.4}$$

expresa la velocidad, donde w es la velocidad angular del marco en el objeto respecto al marco de referencia fijo. Es necesario definir una matriz jacobiana afín J_v adecuada para obtener la siguiente relación

$$v = J_v(d)\dot{d}$$

Como la integral de w no tiene un significado físico claro, los coeficientes de la matriz $J_v(d)$ no tienen relación con las funciones que definen una matriz Jacobiana. Mas adelante se mostrara la matriz jacobiana afin que se usa en este trabajo.

Capítulo 3

Procedimiento

3.1. Estimación de pose

Para resolver el problema de estimación de pose usaremos el algoritmo descrito en [30]. Se toma como referencia una cara plana del objeto, en la cual se requiere conocer las coordenadas de 4 puntos característicos, mismos que se suponen coplanares y que ninguna combinación de 3 de estos puntos sean colineales. Se busca encontrar la pose con respecto al marco de referencia en el que se encuentra la cámara que observa al objeto. Se asume que la pose inicial es conocida, y se nombra π^* al plano del objeto en la posición de referencia conocida; éste tiene acoplado un sistema de coordenadas $\mathcal{F}^*$, como se muestra en la figura 3.1.

Al moverse el objeto, el plano se desplaza a una nueva posición, que llamaremos π, que tiene asociado un marco de coordenadas $\mathcal{F}$. El marco de referencia adjunto a la camara se nombra I. Los 4 puntos característicos del objeto observados desde el marco de referencia I son:

$$\bar{m}_i(t) \triangleq [x_i(t) \quad y_i(t) \quad z_i(t)]^T, \quad i = 1,2,3,4$$

en el plano π y

$$\bar{m}_i^* \triangleq [x_i^* \quad y_i^* \quad z_i^*]^T, \quad i = 1,2,3,4$$

en el plano π^*.

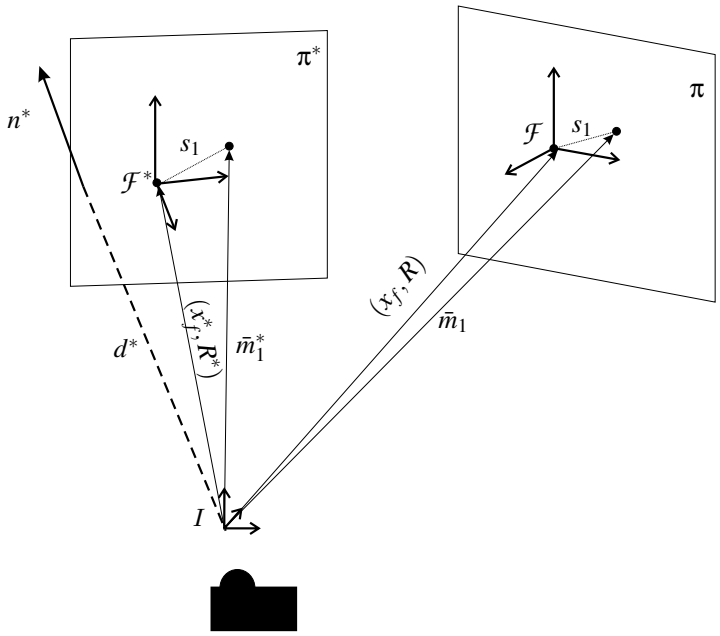

Figura 3.1: Relación entre marcos de referencia

Por las condiciones del problema considerado, en el que los planos son observados por la cámara, asumimos que $z_i(t)$ y z_i^* son siempre positivos, para $i = 1,2,3,4$.

La posición de $\mathcal{F}$ con respecto a I está dada por $R(t)$ y $x_f(t)$, mientras que la posición de $\mathcal{F}^*$ con respecto a I está dada por R^* y x_f^*, y entre $\mathcal{F}$ y $\mathcal{F}^*$ tenemos la relación dada por $\bar{R}$ y $\bar{x}_f(t)$.

Ademas, n^* es el vector normal a π^* y d^* es la distancia constante entre π^* e I

$$d^* = n^{*T} m_i^*,$$

misma que se supone conocida. Además, $\bar{m}_i$ y $\bar{m}_i^*$ se relacionan por [2]

$$m_i = \underbrace{\frac{z_i^*}{z_i}}_{\alpha_i} \underbrace{\left(\bar{R} + \frac{\bar{x}_f}{d^*} n^{*T}\right)}_{H} m_i^* \tag{3.1}$$

Donde m_i son las coordenadas normalizadas de $\bar{m}_i$, y m_i^* son las coordenadas normalizadas de $\bar{m}_i^*$, es decir,

$$m_i(t) \quad \triangleq \quad \frac{\bar{m}_i(t)}{z_i(t)} \quad = \begin{bmatrix} \frac{x_i(t)}{z_i(t)} & \frac{y_i(t)}{z_1(t)} & 1 \end{bmatrix}^T$$

y

$$m_i^* \quad \triangleq \quad \frac{\bar{m}_i^*}{z_i^*} \quad = \begin{bmatrix} \frac{x_i^*}{z_i^*} & \frac{y_i^*}{z_i^*} & 1 \end{bmatrix}^T$$

Ademas, α_i es una relación de profundidad escalar, mientras que $H \in \mathbb{R}^{3x3}$ es una matriz, denominada *matriz de homografía*, que relaciona las proyecciones de los puntos característicos m_i y m_i^* y que depende de la posición relativa entre los planos π y π^*.

Los puntos característicos del objeto son vistos por la cámara con coordenadas proyectadas. Para obtener estas coordenadas se pueden usar diferentes herramientas computacionales existentes; en particular, para este trabajo se utilizaron algoritmos de procesamiento de imágenes de LabView (figura 2.2). Los puntos son obtenidos en la forma

$$p_i^* = [u_i^* \quad v_i^* \quad 1]$$

y

$$p_i = [u_i \quad v_i \quad 1]$$

Estos puntos vienen dados en función de m_i^* y m_i por la relación:

$$\begin{aligned} p_i^* &= Am_i^* \\ p_i &= Am_i, \end{aligned} \tag{3.2}$$

donde A es la matriz de calibración de la cámara [33]. Esta matriz está compuesta por parámetros intrínsecos de la cámara que se está usando. Aunque en ciertas aplicaciones esta matriz se tome, por simplicidad, como la identidad, se debe de considerar que en la práctica las cámaras capturan las imágenes en términos de pixeles; además, el centro de la imagen no siempre se encuentra en el centro focal.

Se puede considerar [34] que la matriz A está compuesta por:

$$A = \begin{pmatrix} S_x & S_\theta & O_x \\ 0 & S_y & O_y \\ 0 & 0 & 1 \end{pmatrix} \tag{3.3}$$

S_x y S_y dependen del tamaño del pixel a lo largo de los ejes x y y respectivamente. O_x y O_y son las coordenadas en pixeles del centro focal. S_θ puede considerarse como un valor de falta de ortogonalidad y depende de θ, el ángulo entre los ejes x y y.

De las ecuaciones (3.1) y (3.2) podemos obtener la relación

$$\begin{aligned} m_i &= A^{-1} p_i \\ A^{-1} p_i &= \alpha_i H m_i^* \\ A^{-1} p_i &= \alpha_i H A^{-1} p_i^* \\ p_i &= A \alpha_i H A^{-1} p_i^* \end{aligned}$$

Nombrando $G = A\alpha_i H A^{-1}$, se obtiene

$$p_i = G p_i^* = \begin{pmatrix} g_{11} & g_{12} & g_{13} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{pmatrix} p_i^* \tag{3.4}$$

A partir del conocimiento las coordenadas de los 4 puntos caracteristicos $p_i(t)$ y sus coordenadas de referencia p_i^*, en [35] se muestra cómo podemos construir un sistema de ecuaciones para calcular la matriz $G(t)$. Usando la ecuación (3.4) para relacionar los puntos p_1 y p_1^*:

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = \begin{pmatrix} g_{11} & g_{12} & g_{13} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{pmatrix} \begin{bmatrix} u_i^* \\ v_i^* \\ 1 \end{bmatrix}$$

$$
\begin{aligned}
u_i &= g_{11}u_i^* + g_{12}v_i^* + g_{13} \\
v_i &= g_{21}u_i^* + g_{22}v_i^* + g_{23} \\
1 &= g_{31}u_i^* + g_{32}v_i^* + g_{33}
\end{aligned}
$$

$$
\frac{u_i}{u_i} = \frac{v_i}{v_i} = g_{31}u_i^* + g_{32}v_i^* + g_{33}
$$

$$
\begin{aligned}
u_i &= g_{31}u_i^*u_i + g_{32}v_i^*u_i + g_{33}u_i \\
v_i &= g_{31}u_i^*v_i + g_{32}v_i^*v_i + g_{33}v_i
\end{aligned}
$$

y se obtienen las siguientes dos ecuaciones

$$
\begin{aligned}
g_{11}u_i^* + g_{12}v_i^* + g_{13} - g_{31}u_iu_i^* - g_{32}u_iv_i^* - g_{33}u_i &= 0 \\
g_{21}u_i^* + g_{22}v_i^* + g_{23} - g_{31}v_iu_i^* - g_{32}v_iv_i^* - g_{33}v_i &= 0
\end{aligned}
\tag{3.5}
$$

para cada par de puntos. Así, obtenemos un sistema de 8 ecuaciones que se resuelven por medio de un algoritmo de mínimos cuadrados.

Con lo anterior se procede a obtener $\alpha H(t) = A^{-1}G(t)A$ a través de la descomposición de $H(t)$, como se muestra en [35], en sus valores singulares de

$$
H(t) = \left(\bar{R} + \frac{\bar{x}_f}{d^*}n^{*T}\right) \tag{3.6}
$$

Usando la descomposición en valores singulares, H se puede descomponer como el producto: $H = U\Lambda V^T$. Donde U y V son matrices ortogonales (que satisfacen $U^TU = V^TV = I$), y Λ es la matriz diagonal compuesta por los valores singulares de H. Se obtienen los valores singulares d_i, los cuales son positivos y se pueden ordenar de manera descendente: $d_1 \geq d_2 \geq d_3$

Usando esta descomposición se obtiene la siguiente ecuación:

$$\Lambda = d'R' + x'_f n'^T$$

R, x_f y n se relacionan con R', x'_f y n' por:

$$\begin{aligned} R &= sUR'V^T \\ X_f &= Ux'_f \\ N &= Vn' \\ d &= sd' \\ s &= \det(U)\det(V) \end{aligned} \tag{3.7}$$

Tomando a $d' = d_2$ se tienen los siguientes casos

- Se calcula la matriz R'

$$\begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix}$$

Donde

$$\begin{aligned} \sin\theta &= \frac{\sqrt{(d_1^2 - d_2^2)(d_2^2 - d_3^2)}}{(d_1 + d_3) + d_2} \\ \cos\theta &= \frac{d_2^2 + d_1 d_3}{(d_1 + d_3)d_2} \end{aligned}$$

También se tiene que

$$x'_f = (d_1 - d_3) = \begin{pmatrix} x_1 \\ 0 \\ -x_3 \end{pmatrix}$$

Donde x_1, x_2 y x_3 son los elementos del vector n':

$$\begin{aligned} x_1 &= \varepsilon_1 \sqrt{\frac{d_1^2 - d_2^2}{d_1^2 - d_3^2}} \\ x_2 &= 0 \\ x_3 &= \varepsilon_3 \sqrt{\frac{d_2^2 - d_3^2}{d_1^2 - d_3^2}} \\ \varepsilon_1, \varepsilon_3 &= \pm 1 \end{aligned}$$

- Cuando se tiene que $x_1 = x_2 = 0$ y $x_3 = \pm 1$, se obtiene

$$\begin{aligned} R' &= I \\ x'_f &= (d_3 - d_1) n' \end{aligned}$$

- En el caso en que x_1, x_2 y x_3 se indefinen, se tiene que:

$$\begin{aligned} R' &= I \\ t' &= 0 \end{aligned}$$

Se obtiene $\bar{R}, \bar{x}_f, d^*$ y n^{*T} usando la relacion (3.7) para despues obtener la rotación con respecto al marco de referencia I

$$R(t) = R^* \bar{R}(t)$$

y calcular la posición del objeto.

$$x_f(t) = x_f^* + R^* \bar{x}_f(t)$$

Una vez que se tiene la rotación entre los planos, podemos usar tal información para

obtener el vector normal al plano del objeto, dado por

$$n = \bar{R}n^*$$

Entonces, para describir la pose del objeto es necesario construir el marco de referencia $\mathcal{F}$, cuyo origen puede fijarse, por conveniencia, en $\bar{m}_1$. Luego se construyen los vectores $i_x, i_y, i_z \in \mathbb{R}^3$ que definen el marco de referencia $\mathcal{F}$ y que forman la matriz $R(t)$ en el marco de referencia I

$$R = [i_x \quad i_y \quad i_z].$$

Las columnas de R son definidas como

$$i_z = -n \tag{3.8}$$

$$i_x = \frac{\bar{m}_2 - \bar{m}_1}{\bar{s}_1} \tag{3.9}$$

$$i_y = -n \times \frac{\bar{m}_2 - \bar{m}_1}{\bar{s}_1} \tag{3.10}$$

Para calcular $\bar{m}_1(t)$ y $\bar{m}_2(t)$, se usa un plano imaginario π', que se muestra en la figura 3.2 y que tiene a $-n$ como vector normal, es decir, es paralelo al plano π. Además, este plano contiene a m_1. Se define una línea l que pasa por el origen de I y que pasa por $m_2(t)$ y $\bar{m}_2$. La línea l intersecta al plano π' en un punto, que será llamado m_2'. En [31] se muestra cómo, usando las primitivas de la línea l y el plano π', los cuales estan definidos por el conjunto de puntos $q \in \mathbb{R}^3$ que satisfacen las funciones implícitas

$$L = \{q \quad | \quad q - um_2 = 0, \forall u \in \mathbb{R}\}, \tag{3.11}$$

$$\pi_x' = \left\{q \quad | \quad n \cdot (q - m_1) = 0,\, q, n, m_1 \in \mathbb{R}^3\right\}, \tag{3.12}$$

que se puede ver que la intersección ocurre en

$$u = \frac{n \cdot m_1}{n \cdot m_2}. \tag{3.13}$$

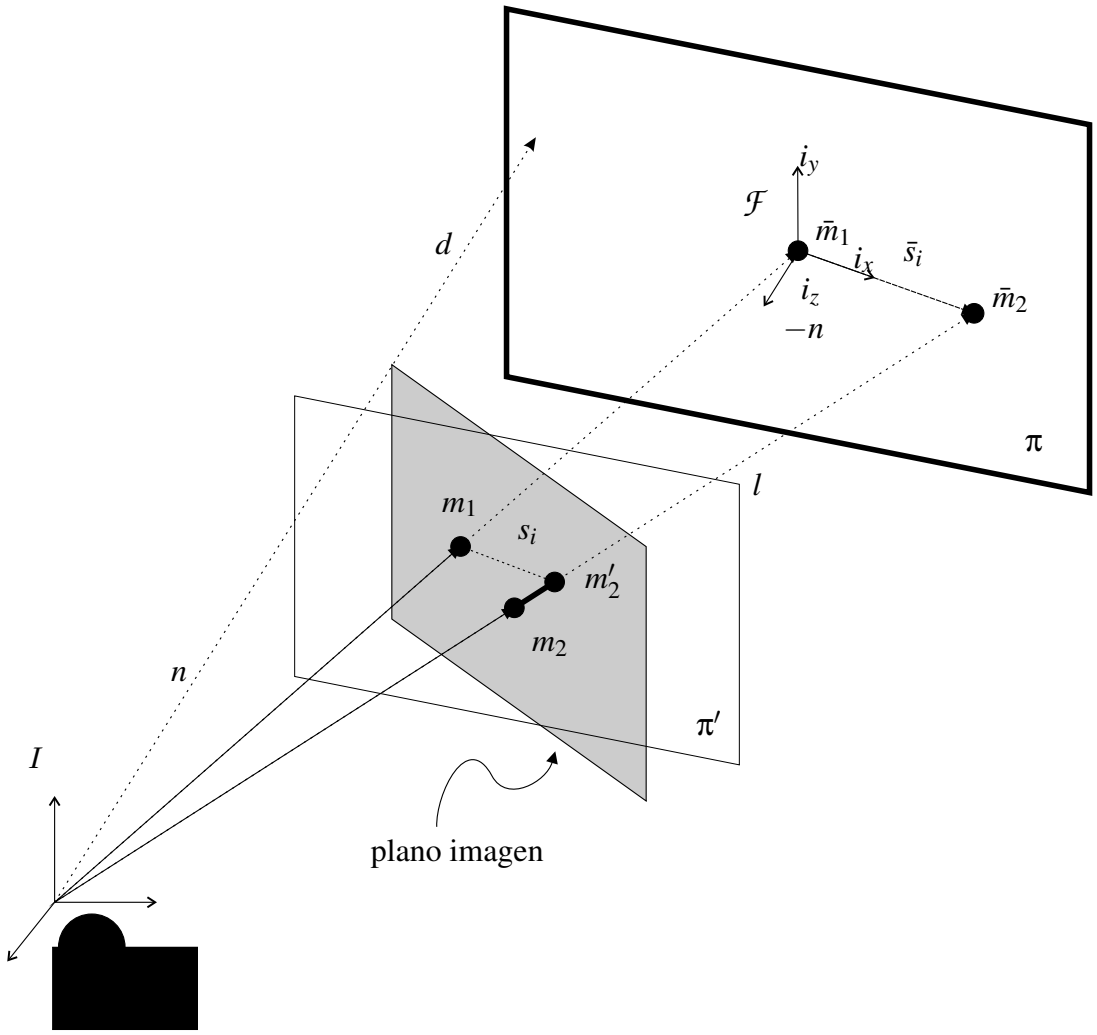

Figura 3.2: En la figura se muestra el uso de un plano auxiliar π', paralelo al plano π, que intersecta en un punto m_2' con la línea l, y que se utiliza para calular los valores faltantes de m_i y $\bar{m}_i$

Si se combinan las expresiones (3.11) y (3.13) se obtiene la ecuación

$$q - \frac{n \cdot m_1}{n \cdot m_2} = 0,$$

a partir de la cual se resuelve para el punto $q = m_2'$

$$m_2' = \frac{n \cdot m_1}{n \cdot m_2}.$$

Una vez que se conoce m_2', se puede obtener s_1

$$s_1 = \|m_2' - m_1\| \tag{3.14}$$

Usando las propiedades de triángulos similares se puede calcular

$$\frac{s_1}{\bar{s}_1} = \frac{m_1}{\bar{m}_1} = \frac{m_2'}{\bar{m}_2}, \tag{3.15}$$

por lo que con ello se logra conocer s_1, $\bar{s}_1$, $\|m_1(t)\|$ y $\|m_2'(t)\|$. Estas ultimas se usan, a su vez, para obtener $\bar{m}_1(t)$ y $\bar{m}_2(t)$

$$\bar{m}_1(t) = \frac{\|\bar{m}_1\|}{\|m_1\|} m_1, \bar{m}_2(t) = \frac{\|\bar{m}_2\|}{\|m_2\|} m_2$$

Ahora las soluciones para i_x, i_y se pueden obtener de las ecuaciones (3.9) y (3.10) y que se pueden usar para obtener $R(t)$. También, como el origen de $\mathcal{F}$ se fijó en $\bar{m}_1(t)$, la traslación está dada simplemente por $x_f(t) = \bar{m}_1(t)$.

Con esto ya se conocen $R(t)$ y $x_f(t)$, las que se usan para definir la pose (2.1) tomando a $R(t)$ como la rotación que buscamos R y a $x_f(t)$ como la traslación buscada t_r. Ya se conoce la pose del objeto observado y que también se usa en el algoritmo de estimación de velocidad, que se explicará mas adelante. Los datos conocidos y que se utlizarán a continuación son:

$\bar{R}(t)$	Rotación entre el plano F^* de referencia y el plano F de la imagen
$\bar{x}_f(t)$	Traslación entre el plano F^* de referencia y el plano F de la imagen
p_1^*	Punto característico de referencia, observado desde la cámara
p_1	Punto característico al moverese el objeto, observado desde la cámara

3.2. Estimación de velocidad

Para calcular la velocidad del objeto usamos el observador descrito en [2]. Primero se obtiene la dinámica de un objeto en movimiento, definiendo un error de posición del objeto

$$e = [e_v \quad e_w]^T$$

donde e_v es la diferencia de posicion de los planos π y π^*

$$e_v = P_e(t) - P_e^*$$

Donde

$$P_e = [u_i \quad v_i \quad \ln(z_i)]$$

$$P_e^* = [u_i^* \quad v_i^* \quad \ln(z_i^*)]$$

y e_w es la diferencia de posición dada por la rotación,

$$e_w = \theta(t)U(t) \tag{3.16}$$

que es la representación llamada "eje ángulo" de una rotación. Donde $U(t) \in \mathbb{R}^3$ representa un vector unitario que funciona como eje de rotación, y $\theta(t) \in \mathbb{R}$ es el ángulo de rotación alrededor de $U(t)$, con la suposición de que el ángulo está acotado por:

$$-\pi < \theta(t) < \pi \tag{3.17}$$

Esta representación de la rotación no es única, como se muestra en [36]. Esto es debido a que $-\theta(t)$ alrededor de $-U(t)$ representa la misma rotacion que $\theta(t)$ alrededor de $U(t)$. Una solución particular está dada por [2]

$$\theta_p(t) = \cos^{-1}\left(\frac{1}{2}(\mathrm{tr}(\bar{R} - 1))\right)$$

y para obtener el vector $U_p(t)$, obtenemos una representación en matriz antisimétrica partic-

ular dada por

$$[u_p]_x = \frac{\bar{R} - \bar{R}^T}{2\sin\theta_p}$$

$$0 \leq \theta_p \leq \pi$$

Como θ_p tiene mayor restricción que θ se puede extender el resultado para

$$\theta_p U_p = \theta U.$$

Derivando e_v se obtiene que

$$\dot{e}_v = \dot{p}_e = \frac{\alpha_1}{z_1^*} A_e L_v [v_e - \bar{R}[s_i]_x \bar{R}^T w_e]$$

donde

$$A_e = A - \begin{pmatrix} 0 & 0 & u_0 \\ 0 & 0 & v_0 \\ 0 & 0 & 0 \end{pmatrix}$$

debido a que los términos constantes se eliminan al derivar, y

$$L_v = -\begin{pmatrix} 1 & 0 & -\frac{x_1}{z_1} \\ 0 & 1 & -\frac{y_1}{z_1} \\ 0 & 0 & 1 \end{pmatrix}.$$

Si derivamos e_w obtendremos

$$\dot{e}_w = L_w w_e$$

Donde

$$L_w = I_3 - \frac{\theta}{2}[u]_x + \left(1 - \frac{\text{sinc}(\theta)}{\text{sinc}^2(\theta/2)}\right)[u]_x^2$$

$$\text{sinc}(\theta) \triangleq \frac{\sin(\theta)}{\theta}.$$

De lo anterior tenemos que la derivada del error de posición del objeto está dada por

$$\dot{e} = Jv \tag{3.18}$$

donde

$$J = \begin{pmatrix} \frac{\alpha_1}{z_1^*} A_e L_v & \frac{\alpha_1}{z_1^*} A_e L_v \bar{R} [s_i]_x \bar{R}^T \\ 0 & L_w \end{pmatrix} \tag{3.19}$$

que describe la cinemática de un objeto rígido en movimiento.

En [2] se considera una *señal de error de observación*

$$\bar{e} = e - \hat{e} \tag{3.20}$$

donde $\hat{e}$ es generado por

$$\dot{\hat{e}} = \int_{t_0}^{t} (K + I_6)\bar{e}(\tau)d\tau + \int_{t_0}^{t} \rho \text{sign}(\bar{e})d\tau + (K + I_6)\bar{e}(t) \tag{3.21}$$

Donde $K \in \mathbb{R}^{6x6}$ es una matriz diagonal, positiva, de ganancias. Ádemas, $\rho \in \mathbb{R}$ es una constante positiva e $I_6 \in \mathbb{R}^{6x6}$ denota a la matriz identidad de 6×6.

Haciendo un analisis de estabilidad, en [2] se demuestra que $\bar{e} \to 0$ cuando $t \to \infty$, por lo que tambien podemos concluir que $\hat{e} \to e$, y esto nos permite usar (3.18) para calcular la velocidad de nuestro objeto a partir de

$$v = J^{-1}\hat{e}. \tag{3.22}$$

A continuación se muestra dicho análisis presentado en [2].

Para facilitar el análisis se define un error de observación filtrado denotado por $r(t) \in \mathbb{R}^6$

$$r(t) = \dot{\bar{e}}(t) + \bar{e}(t) \tag{3.23}$$

Después de derivar con respecto al tiempo, se obtiene la expresión

$$\dot{r}(t) = \ddot{e}(t) - \dddot{\hat{e}}(t) + \dot{\hat{e}}(t)$$

El observador (3.21) logra el objetivo de control si la ganancia del observador ρ cumple con

$$\rho > \beta_1 + \beta_2 \tag{3.24}$$

Donde β_1 y β_2 son cotas de $\|\ddot{e}(t)\|$ y de $\|e^{(3)}(t)\|$ respetivamente. Para probar esto, se definió una función $V(t) \in \mathbb{R}$

$$V(r,\hat{e},t) \triangleq \frac{1}{2} r^T r + \frac{1}{2}\hat{e}^T \hat{e} + P \tag{3.25}$$

Donde se usa una función auxiliar $P(t) \in \mathbb{R}$ y se define como

$$P(t) \triangleq \varsigma_b - \int_{t_o}^{t} L(\tau) d\tau \tag{3.26}$$

y $\varsigma_b, L(t) \in \mathbb{R}$ que se definen de la siguiente manera

$$\varsigma_b \triangleq p \sum_{i=1}^{6} \left| e_i^{-T}(t_0) \right| - e^{-T}(t_0)\ddot{e}(t_0) \quad L \triangleq r^T [\ddot{e} - \rho \text{sign}(\bar{e})]. \tag{3.27}$$

La función auxiliar $P(t)$ es no negativa cuando ρ cumple con la condición (3.24) . Derivando V se obtiene

$$\dot{V}(r,\hat{e},t) = r^T [\ddot{e} - (K + I_6)r - \rho \text{sign}(\hat{e}) + \dot{\hat{e}}] + \dot{\hat{e}}^T \bar{e} - L(t) \tag{3.28}$$

la que, usando (3.27) y (3.23), se puede reescribir como:

$$\dot{V} = -(K + I_6) \| r \|^2 + \| r \|^2 - \| \bar{e} \|^2 \tag{3.29}$$

y simplificando, la ecuación queda

$$\dot{V} = -K \| r \|^2$$

Donde $r(t), \bar{e}(t), P(t) \in L_\infty$ y $r(t) \in L_2$ [37], y se puede determinar que $\dot{\bar{e}} \in L_\infty$ [37]. Asumiendo que $e(t), \dot{e}(t) \in L_\infty$, se determina que $\hat{e}(t), \dot{\hat{e}}(t) \in L_\infty$, y de la misma manera $\dot{r}(t) \in L_\infty$. Basados en el hecho de que $r(t) \in L_\infty \cap L_2$ y $\dot{r} \in L_\infty$, en [38] se demuestra, usando el Lema de Barbalat, que

$$\lim_{t \to \infty} \| r(t) \| = 0. \tag{3.30}$$

Y de la ecuación (3.20) se sigue que $\lim_{t \to \infty} \| \bar{e}(t) \| = 0$ y, por lo tanto, $\hat{e}(t) \to e(t)$ conforme $t \to \infty$.

3.3. Programa computacional

Se realizó un programa mostrado en la figura 3.3, basándonos en el algoritmo anteriormente descrito, en el cual se incluyen tanto el algoritmo de estimación de pose [30], como el observador para calcular la velocidad [2]. En la figura 3.3 se muestra el diagrama de bloques del progama.

Comenzando desde arriba, la primera parte es la función *simrotación* que consta de un programa que simula el movimiento de 4 puntos en un plano vistos desde una cámara. El programa tiene como entrada una rotación y traslación deseadas, dependientes del tiempo, además del tiempo de simulación. A partir de éstos, la función *simrotación* genera una matriz de homografía $H(t)$, usando la relación (3.6). Una vez que se tiene la matriz de homografía, se proyectan los puntos de su posición inicial p_i^* a la posición $p_i(t)$, con la rotación y traslación deseadas en la simulación. Esta primera función del programa tiene como salida las coordenadas proyectadas de los puntos (3.2).

A continuación, el programa calcula la pose del objeto simulado por los puntos generados anteriormente. La función *estrotacion* primero obtiene la matriz $H(t)$, y utilizando el metodo de minimos cuadrados resuelve el sistema de ecuaciones creado a partir de las coordenadas de los puntos, para después descomponerla en sus componentes utilizando las condiciones dadas en [35] y descritas anteriormente. Con lo anterior, se obtiene $R(t)$ y $x_f(t)$. Esta función

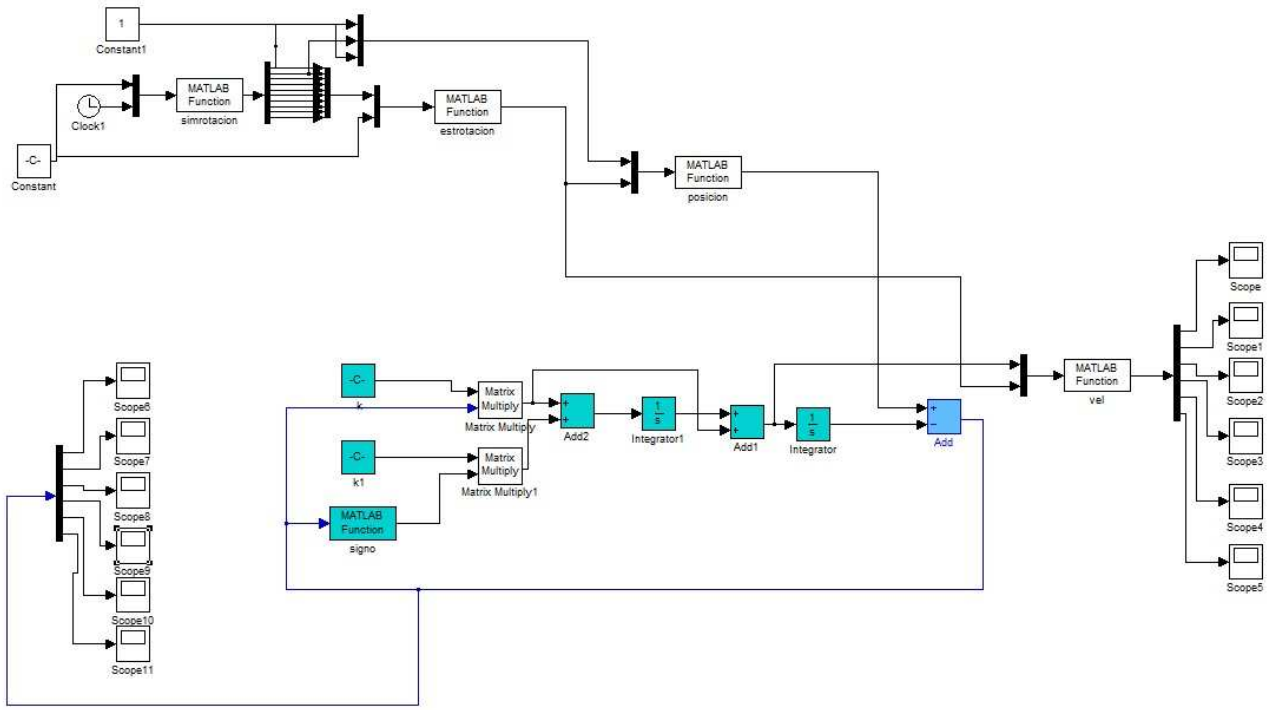

Figura 3.3: Programa de estimación en simulink

tiene como salida la rotación y traslación del objeto simulado.

La siguiente parte del programa es el observador. La función *posicion* calcula $e(t)$. e_v se obtiene de la diferencia en las coordenadas producidas por *simrotación*. e_w se obtiene de la rotación calculada por *estrotacion*; estas variables constituyen la entrada al observador. Una vez que ha calculado $\hat{e}$ en los bloques que representan la ecuación (3.21), $\hat{e}$ se conecta como entrada a la función *vel*, que calcula la matriz J utilizando la ecuacion (3.19), a partir de $R(t)$ calculada en *estrotacion* y de las constantes anteriormente descritas, entrega como salida la velocidad obtenida de (3.22).

Listado de funciones utilizadas en el algoritmo de estimación de pose.

homografialsqr	calcula la matriz H tomando 4 pares de puntos utilizando mínimos cuadrados
descomposicion	descompone la matriz H, usando la descomposición SVD mostrada en [35].
“posesinm”	calculan la matriz R del objeto con respecto al marco de la cámara[31].

Listado de funciones utilizados en el algoritmo de estimación de velocidad.

simrotacion	Simula el movimiento de puntos característicos, desde el punto de vista de una camara.
estrotacion	Calcula la rotación utilizando el algoritmo de estimación de pose.
posicion	Calcula el error de posición, comparando los puntos simulados contra la posición de referencia
vel	Calcula la veocidad del objeto utilizando la matriz Jacobiana afín

Capítulo 4

Simulaciones

Para probar el algoritmo, se desarrolló un programa que simula los 4 puntos característicos de un plano en movimiento. El simulador toma en cuenta los efectos de la perspectiva y tiene como salida la posición de los puntos como si fueran observados desde una cámara, es decir, en 2 dimensiones y con sus coordenadas en pixeles. En el simulador se pueden programar la rotación y traslación deseada del objeto simulado.

Para probar la estimación de pose, se probaron a manera de ejemplo los siguientes movimientos

Se simuló una rotación de

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0{,}6428 & -0{,}7660 \\ 0 & 0{,}7660 & 0{,}6428 \end{bmatrix}$$

$$t = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

que representa una rotacion de 50° alrededor de eje x, como se muestra en la figura 4.1.

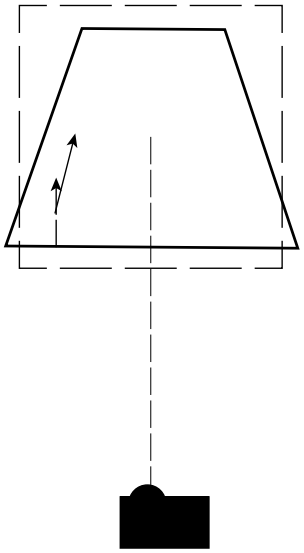

Figura 4.1: Rotación sobre el eje x

El algoritmo de estimación obtuvo

$$R = \begin{bmatrix} -1{,}0000 & -0{,}0000 & 0{,}0000 \\ 0{,}0000 & 0{,}6910 & -0{,}7241 \\ -0{,}0000 & 0{,}7241 & 0{,}6910 \end{bmatrix}$$

$$t = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Se puede ver que existe un ligero error en la estimación. Ésto debido al algoritmo de estimación de la matriz de homografía, el cual, como se explicó anteriormente, se realiza utilizando un método de mínimos cuadrados, el cual nos da una aproximacion que para este caso no es exacta. Dicho error se ve reflejado en la descompición de la matriz para obtener R.

También se simuló

$$R = \begin{bmatrix} 0{,}6428 & 0{,}7660 & 0 \\ -0{,}7660 & 0{,}6428 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$t = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

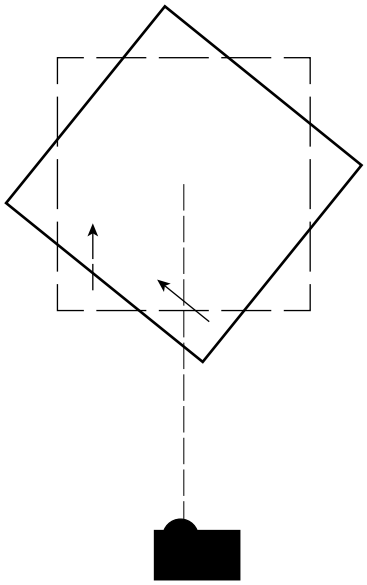

Figura 4.2: Rotación sobre el eje z

en este se caso se trata de una rotación de 50° alrededor del eje z, representada en la figura 4.2. Y el algoritmo de estimación obtuvo

$$R = \begin{bmatrix} 0{,}6428 & 0{,}7661 & -0{,}0003 \\ -0{,}7661 & 0{,}6428 & -0{,}0000 \\ 0{,}0002 & 0{,}0002 & 1{,}0000 \end{bmatrix}$$

$$t = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Se puede ver que para este caso la aproximación es bastante más cercana al valor real.

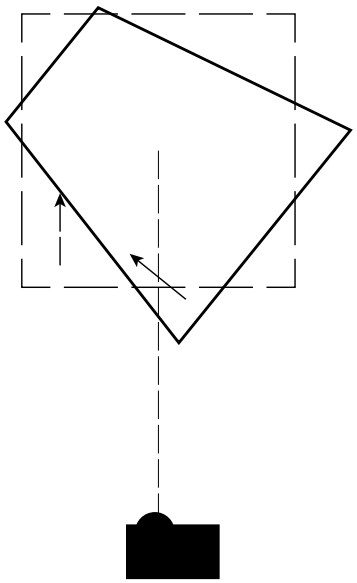

Figura 4.3: Rotación compuesta por una rotación sobre x y una rotación sobre z

Otro ejemplo es la combinación de ambas rotaciones, esto es, 50° alrededor de x y 50° alrededor de z. Dicha rotación se puede ver representada en la figura 4.3. Se tiene que la matriz que representa tal rotación es:

$$R = R_x * R_z = \begin{bmatrix} 0{,}6428 & 0{,}4924 & -0{,}5868 \\ -0{,}7660 & 0{,}4132 & -0{,}4924 \\ 0 & 0{,}7660 & 0{,}6428 \end{bmatrix}$$

y la Rotación calculada por el estimador de pose fue

$$R = \begin{bmatrix} 0{,}6428 & 0{,}4598 & -0{,}7342 \\ -0{,}7660 & 0{,}3858 & -0{,}3415 \\ 0{,}0003 & 0{,}7998 & 0{,}5868 \end{bmatrix}$$

En este ejemplo tambien existe un error, proveniente de la estimación de la matriz de homografía.

El mismo simulador luego se incorporó al estimador de velocidad. Se introdujo una rotación alrededor de x y traslación a lo largo de los ejes y y z, ambos variantes en el tiempo. Para probar el estimador se graficó la señal del error (3.20), que, como se mencionó anteriormente, debe tender a 0.

A continuación se muestran las gráficas de cada una de las componentes de la velocidad.

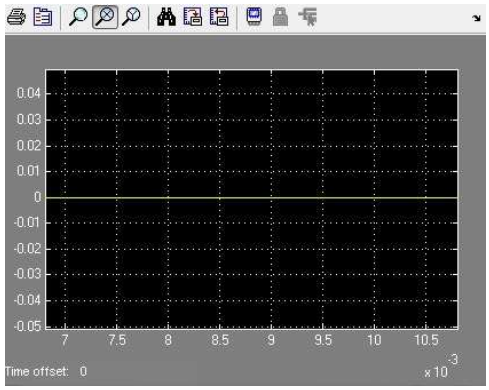

Figura 4.4: Error en la velocidad rotacional sobre el eje x

En la figura 4.4 vemos que no existe error al ser la rotación alrededor de este eje.

Para el caso del error en los ejes y y z, se puede ver en las figuras 4.5 y 4.6, que el error tiende a 0.

En cuanto al error en la traslación podemos ver que, a pesar de tener algunos picos, el error es muy próximo a 0.

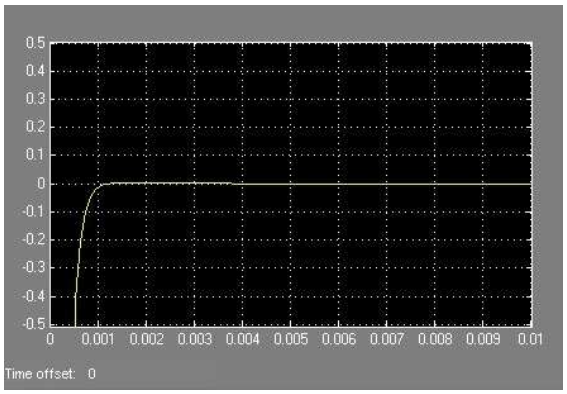

Figura 4.5: Error en la velocidad rotacional sobre el eje *y*

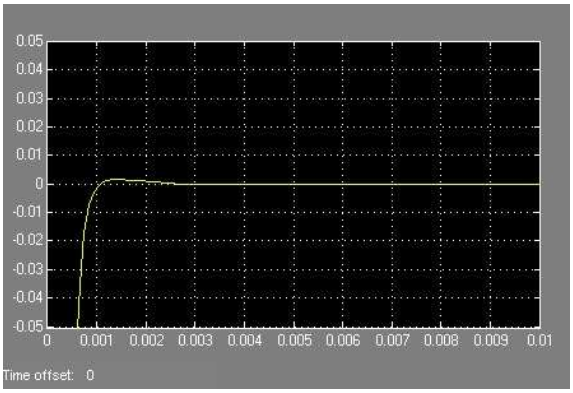

Figura 4.6: Error en la velocidad rotacional sobre el eje *z*

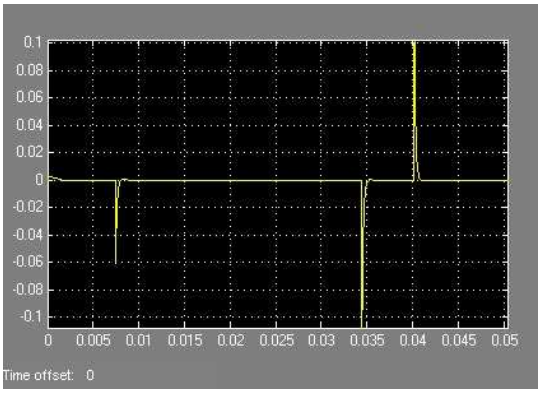

Figura 4.7: Error en la velocidad traslacional en *x*

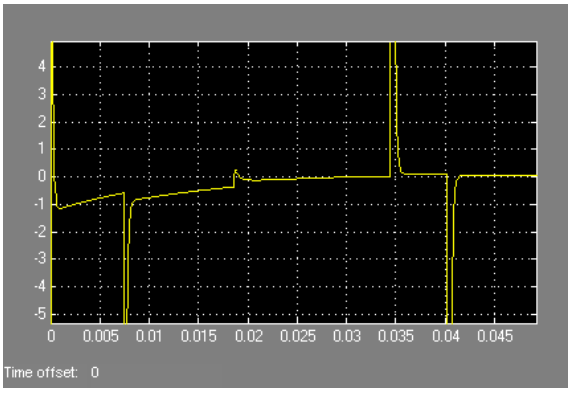

Figura 4.8: Error en la velocidad traslacional en y

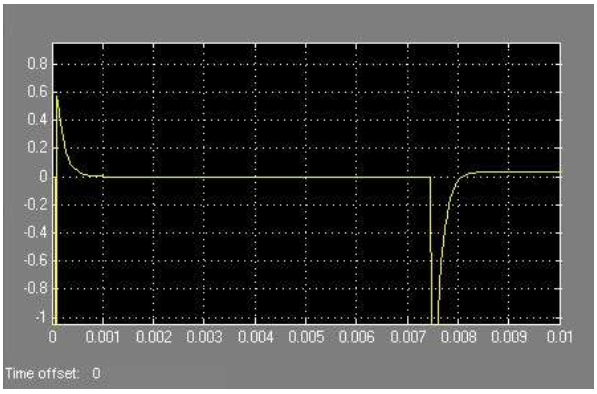

Figura 4.9: Error en la velocidad traslacional en z

Capítulo 5

Conclusiones

A lo largo de este trabajo de tesis se logró mostrar la integración de los algoritmos de estimación de pose y velocidad, mostrados en [30], se adecúan al escenario planteado. Se describieron los algoritmos y se probaron con ejemplos sencillos que se aproximan a la situación planteada.

Del mismo modo, en el Capítulo 2 se presentaron otros algoritmos que abordan el problema de manera diferente y que tienen otros requerimientos, lo que nos muestra que en el área de visión computacional se siguen desarrollando nuevas técnicas, mismas que buscan adaptarse a escenarios cada vez más complejos.

Cabe recordar los elementos decisivos para la elección del algoritmo. La observación de al menos 4 puntos característicos cooplanares, el conocimento de la distancia entre al menos dos de esos puntos, los cuales se cumplen en el escenario planteado. Y otro aspecto es el uso de una sola cámara como sensor, para la obtención de la información requerida.

Se ilustró que el calcular todos los componentes de la velocidad de objeto moviéndose en el espacio no es un problema trivial.

Como conclusiones generales tenemos que: Es posible calcular la pose de un objeto observado desde una sola cámara y usar esta información para calcular la velocidad de dicho

objeto.

Se puede mejorar la precisión del algoritmo con el uso de otros métodos para calcular la matriz de homografía. Por ejemplo, el uso de un filtro de Kalman.

El algoritmo mostrado se puede aplicar para otros escenarios más complejos que cumplan con las suposiciones establecidas. Que el objeto observado cuente con una cara plana, se obtengan cuatro puntos característicos coplanares y no colineales y se conozca la distancia entre dos de estos puntos característicos, para calcular la pose. Y que se conozcan las coordenadas de un punto en el objeto y su pose, para estimar la velocidad.

Bibliografía

[1] B. Ghosh, H. Inaba, and S. Takahasbi, "Identification of riccati dynamics under perspective and orthographic observations," *Automatic Control, IEEE Transactions on*, vol. 45, pp. 1267–1278, Jul 2000.

[2] V. Chitrakaran, D. Dawson, W. Dixon, and J. Chen, "Identification of a moving object's velocity with a fixed camera," in *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, vol. 5, pp. 5402–5407 Vol.5, 2003.

[3] B. Ghosh and E. P. Loucks, "A realization theory for perspective systems with applications to parameter estimation problems in machine vision," *Automatic Control, IEEE Transactions on*, vol. 41, pp. 1706–1722, Dec 1996.

[4] K. Hashimoto and T.Ñoritsugu, "Visual servoing with linearized observer," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 1, pp. 263–268 vol.1, 1999.

[5] B. Ghosh, H. Inaba, and S. Takahasbi, "Identification of riccati dynamics under perspective and orthographic observations," *Automatic Control, IEEE Transactions on*, vol. 45, pp. 1267–1278, Jul 2000.

[6] A. Koivo and N. Houshangi, "Real-time vision feedback for servoing robotic manipulator with self-tuning controller," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 21, pp. 134–142, Jan 1991.

[7] P. Allen, A. Timcenko, B. Yoshimi, and P. Michelman, "Trajectory filtering and prediction for automated tracking and grasping of a moving object," in *Robotics and Au-*

tomation, 1992. Proceedings., 1992 IEEE International Conference on, pp. 1850–1856 vol.2, May 1992.

[8] G. Agin and T. Binford, "Computer description of curved objects," *Computers, IEEE Transactions on*, vol. C-25, no. 4, pp. 439–449, 1976.

[9] R.Ñevatia and T. O. Binford, "Description and recognition of curved objects," *Artificial Intelligence*, vol. 8, no. 1, pp. 77–98, 1977.

[10] D. Marr, *Vision - A computational investigation into the human representation and processing of visual information An invitation to Cognitive Science*. San Francisco: W. H. Freeman, 1982.

[11] A. P. Witkin, "Scale-space filtering," in *Proceedings, 8th International Joint Conference on Artificial Intelligence*, pp. 1019–1022, August 1983. Karlsruhe.

[12] A. Witkin, D. Terzopoulos, and M. Kass, "Signal matching through scale space," *International Journal of Computer Vision*, pp. 714–719, 1987.

[13] T. Lindeberg, "Scale-space for discrete signals," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 12, no. 3, pp. 234–254, 1990.

[14] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes - Active Contour Models," *International Journal Of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1987.

[15] M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking," 1998.

[16] A. Lanitis, C. Taylor, and T. Cootes, "Automatic interpretation and coding of face images using flexible models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 19, no. 7, pp. 743–756, 1997.

[17] I. Matthews, J. Xiao, and S. Baker, "2d vs. 3d deformable face models: Representational power, construction, and real-time fitting.," *International Journal of Computer Vision*, vol. 75, no. 1, pp. 93–113, 2007.

[18] H. Sidenbladh and M. J. Black, "Learning the statistics of people in images and video," vol. 54, pp. 181–207, Sept. 2003.

[19] A. Blake and M. Isard, *Active Contours*. Springer Verlag, 1998.

[20] R. Fergus, P. Perona, and A. Zisserman, "A sparse object category model for efficient learning and complete recognition," in *Toward Category-Level Object Recognition* (J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, eds.), vol. 4170 of *LNCS*, pp. 443–461, Springer, 2006.

[21] R. Haralick, H. Joo, D. Lee, S. Zhuang, V. Vaidya, and M. Kim, "Pose estimation from corresponding point data," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 19, no. 6, pp. 1426–1446, 1989.

[22] S. Lavallee and R. Szeliski, "Recovering the position and orientation of free-form objects from image contours using 3d distance maps," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 17, no. 4, pp. 378–390, 1995.

[23] S. Lee and Y. Kay, "An accurate estimation of 3-d position and orientation of a moving object for robot stereo vision: Kalman filter approach," in *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pp. 414–419 vol.1, 1990.

[24] J. Wang and W. J. Wilson, "3d relative position and orientation estimation using kalman filter for robot control," in *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pp. 2638–2645 vol.3, 1992.

[25] V. Lippiello, B. Siciliano, and L. Villani, "Position and orientation estimation based on kalman filtering of stereo images," in *Control Applications, 2001. (CCA '01). Proceedings of the 2001 IEEE International Conference on*, pp. 702–707, 2001.

[26] B. K. P. Horn, "Relative orientation," *International Journal of Computer Vision*, pp. 59–78, 1990.

[27] D. Oberkampf, D. DeMenthon, and L. Davis, "Iterative pose estimation using coplanar points," in *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR '93., 1993 IEEE Computer Society Conference on*, pp. 626–627, 1993.

[28] Y. Ejiri and N. Hamada, "Moving object velocity estimation using feature point image," in *TENCON 2006. 2006 IEEE Region 10 Conference*, pp. 1–3, 2006.

[29] A. A. H. Ab-Rahman, U. Sheikh, M. Maliki, R. Heriansyah, K. Singh, and S. A. R. Abu-Bakar, "Vestro: Velocity estimation using stereoscopic vision," in *Computers, Communications, Signal Processing with Special Track on Biomedical Engineering, 2005. CCSP 2005. 1st International Conference on*, pp. 120–124, 2005.

[30] A. Dani, S. Velat, C. Crane, N. Gans, and W. Dixon, "Experimental results for image-based pose and velocity estimation," in *Control Applications, 2008. CCA 2008. IEEE International Conference on*, pp. 1159 –1164, sept. 2008.

[31] N. Gans, A. Dani, and W. Dixon, "Visual servoing to an arbitrary pose with respect to an object given a single known length," in *American Control Conference, 2008*, pp. 1261–1267, 2008.

[32] T. Yoshikawa, *Foundations of Robotics: Analysis and Control*. MIT Press, 1990.

[33] T. Yoshikawa, *An Invitation to 3-D Vision*. Interdisciplinary Applied Mathematics, Springer, 2004.

[34] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second ed., 2004.

[35] O. Faugeras and F. Lustman, "Motion and structure from motion in a piecewise planar environment," Tech. Rep. RR-0856, INRIA, June 1988.

[36] M. Spong, *Robot Dynamics And Control*. Wiley India Pvt. Limited, 2008.

[37] D. M. D. W. E. Dixon, A. Behal and S.Ñagarkatti, *Non.linear Control of Engineering Systems: A Lyopunov-Based Approach*. Control Engineering, Birkhöuser, 2003.

[38] J.-J. E. Slotine, W. Li, *et al.*, *Applied nonlinear control*, vol. 199. Prentice hall New Jersey, 1991.