Proyecto PISALA: Sensorización Inteligente para el Seguimiento de Líneas por Visión Artificial

V. Girbés E. Solanes E. Martínez L. Armesto J. Tornero
Universidad Politécnica de Valencia, Inst. de Diseño y Fabricación (IDF), Camino de Vera s/n, 46022, Valencia
{vigirjua@etsii.upv.es, juasogal@upvnet.upv.es; enmarbe1@etsii.upv.es; leoaran@isa.upv.es; jtornero@isa.upv.es}

Resumen- **En este artículo se presenta un sistema embebido para la detección de líneas por visión artificial y su seguimiento mediante una unidad PTU (Pan-Tilt Unit). Así mismo, se muestran diversos resultados experimentales tanto para el proceso de detección de la línea en la imagen como para el propio seguimiento de la misma. El sistema tiene aplicación dentro del Proyecto de Investigación PISALA, cuyo objetivo es que un AGV completamente automatizado siga una línea pintada en el suelo. Dicho proyecto tiene un claro interés industrial en la consecución de almacenes inteligentes.**

I. INTRODUCCIÓN

Hoy por hoy, existen diversas soluciones comerciales con vehículos auto-guiados o AGVs (Auto-Guided Vehicles), véase por ejemplo [1,2,3] entre otras. Estos vehículos han sido tradicionalmente guiados mediante imanes, cables o laser ubicados en zonas específicas del almacén. Otro tipo de guiado está basado en navegación inercial, en la que acelerómetros y giróscopos, junto con *transponders* embebidos en el suelo permiten localizar la posición del vehículo. El mayor inconveniente que presentan los AGVs tradicionales es su poca flexibilidad para adaptarse a los cambios del entorno, el coste asociado a la infraestructura que requieren, en ocasiones no pueden replanificar su ruta, etc.

Con objeto de aumentar el grado de autonomía de los AGV, los investigadores han desarrollado en las últimas décadas numerosas herramientas para resolver los problemas que se presentan tradicionalmente en la robótica móvil, tales como, simulación [4], la localización y la construcción de mapas (SLAM) [5], navegación y planificación de trayectorias [6, 7, 8], coordinación entre robots, etc. En este sentido, se requiere cada vez más sensores con mayor capacidad de procesamiento y abstracción, como por ejemplo los sistemas de visión, que están adquiriendo una importancia cada vez mayor debido a la continua mejora de la tecnología. Tómese como ejemplo la aplicación ROBOLIFT© [9], en la que se utiliza un sistema de visión para detectar marcas artificiales en el suelo. En [10] y [11], se resuelve el problema maniobras de carga y descarga con pallets, utilizando para ello un sistema de visión.

En [12, 13, 14], se desarrollaron varias aplicaciones industriales con AGVs, en las que fundamentalmente se realizaron diversas propuestas de automatización, definiendo arquitecturas hardware y software, orientadas a la tele-operación de vehículos, la navegación autónoma o el seguimiento de línea por visión artificial.

Este artículo se enmarca dentro del contexto de los AGV en el que se describe una aplicación de interés industrial dentro del sector de la Logística. En el artículo se presentan algunos resultados intermedios del proyecto PISALA (Prototipo Industrial para el Seguimiento Automático de Líneas por Visión Artificial). Se trata de un primer proyecto de Investigación en el que se exploran y desarrollan diversas soluciones para la automatización integral de almacenes.

En el artículo describen los resultados obtenidos dentro del contexto de una aplicación para el seguimiento de línea por visión artificial, en cuyo caso, el objetivo es que el AGV siga un circuito pintado en el suelo utilizando un sistema de visión embebido. La idea conceptual del proyecto se muestra en la Figura 1.

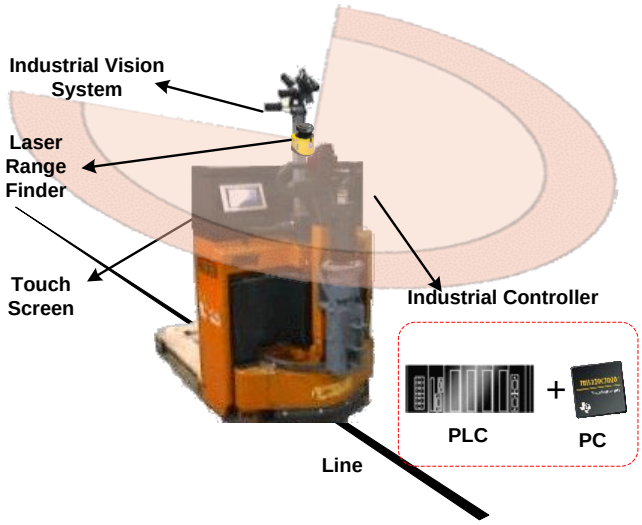


Figura 1. Idea conceptual del Proyecto PISALA.

En la sección II se realiza una descripción de la plataforma, incluyendo los elementos de automatización, sensorización y control incorporados al vehículo, así como los módulos software desarrollados. En la sección III, se describe la algorítmica necesaria para la detección de líneas en imágenes utilizando el sistema de visión embebido CMUCAM3, el cálculo de los parámetros geométricos que describen la línea, así como el conjunto de puntos que pertenecen a ella. La matemática necesaria para proyectar puntos en la imagen al suelo, con objeto de deshacer de la perspectiva queda descrita en la sección VI, mientras que en la sección V se propone un sencillo algoritmo de seguimiento de línea (tracking). Finalmente, en la sección VI se extraen las conclusiones del artículo y se discuten los trabajos futuros.

Trabajos Relacionados

En [15], desarrollaron un sistema en el que un vehículo es capaz de seguir una línea blanca en el suelo, así como un algoritmo de transición del estado que les permite detectar bifurcaciones mediante reconocimiento de patrones. En [16] se utiliza un sistema de visión junto con un RADAR para producir imágenes de rango en un vehículo militar. El objetivo es detectar los bordes de la carretera para que el vehículo se mueva de forma autónoma. En el ámbito de la agricultura, podemos encontrar diversas soluciones de vehículos agrícolas autónomos o semi-autónomos con capacidad de detectar la propia trazada del sesgado o las líneas de cultivo, véase [17] a modo de ejemplo. De forma similar en [18] se describe a un robot con capacidad de eliminar la pintura de barcos siguiendo su propia trazada.

II. DESCRIPCIÓN DE LA PLATAFORMA

El vehículo industrial utilizado es una transpaleta modelo OMG218K (véase Figura 1), se trata desde un punto de vista cinemático de un vehículo con configuración triciclo. Dispone por tanto de dos ruedas traseras no motorizadas de apoyo y una rueda delantera que incorpora un motor de tracción.

La configuración original del vehículo no se disponía de ningún tipo de actuación para la columna de dirección, con lo que se diseñó una columna de dirección, incorporándole un motor para su completa automatización. En la actualidad, ambos motores están siendo controlados por un PLC, modelo CJ1M-CPU21, que es el encargado de realizar de adquirir y procesar todas las señales eléctricas de más bajo nivel, satisfaciendo los requisitos de seguridad y robustez requeridos en esta aplicación.

En concreto, el PLC realiza el control de velocidad sobre la rueda (tracción) utilizando un regulador LQG y el control de posición (angular) para la columna de dirección con un regulador PD. Así mismo, el PLC permite obtener la lectura de dos encoders que determinan la velocidad de giro de la rueda delantera y el ángulo de la misma. La subrutina de control ha sido implementada en una única interrupción temporizada (10ms de periodo de muestreo. El PLC recibe los comandos de referencias a través de la interfaz Hostlink con comunicación serie RS-232.

Como principales elementos de sensorización, el vehículo dispone de un láser de telemetría de SICK, modelo S200 y un sistema de visión embebido CMUCAM3. El láser de telemetría cumple una doble función ya que por un lado permite detectar los objetos más próximos al vehículo con objeto de evitar la colisión, mientras que por otro lado se utiliza para localizar la posición del vehículo y construir un mapa del entorno de forma dinámica. El sensor CMUCAM3 [19] desarrollado por la Carnegie Mellon University, es un sistema embebido de visión que permite implementar algoritmos de visión artificial, es completamente programable y está basado en el procesador ARM7TDMI. Se ha montado el sistema de visión sobre una torreta que realiza las funciones de una PTU (Pan-Tilt Unit) y que es directamente controlada por la propia CMUCAM3 mediante dos servomotores.

III. DETECCIÓN DE LÍNEAS EN IMÁGENES

En esta sección se describe el algoritmo para la detección de líneas y obtención de sus parámetros característicos con el fin de realizar un seguimiento de línea con un vehículo auto-guiado. El algoritmo ha sido específicamente diseñado para poder procesar una imagen en escala de grises de 176x144 pixeles, que corresponde al modo de baja resolución con formato YCrCb (sólo se trabaja con el canal Y). La CMUCAM3 tiene una limitación importante en cuanto a capacidad de RAM disponible que en la práctica hace que podamos trabajar sólo con 3 filas como máximo, condicionando significativamente la implementación del algoritmo realizada. Se parte de las siguientes premisas de partida:

- La orientación de la línea es más bien vertical, pudiéndose llegar a detectar líneas prácticamente en la horizontal, aunque con mayor dificultad.
- El fondo de la imagen puede contener suciedad (por ejemplo una mancha de aceite) u objetos que puedan perturbar la percepción de la línea, si bien el fondo y la línea deben estar lo suficientemente contrastados.
- Se pueden disponer en una misma imagen de varias líneas, ya sea por bifurcaciones, tramos curvos, etc.

A continuación se muestra el pseudocódigo y una descripción detallada de cada uno de sus pasos:

1. Para cada fila, determinar los bordes de la línea basándose en la diferencia de intensidad de los pixeles vecinos (búsqueda por la izquierda y la derecha de la imagen).
2. Repetir el proceso los pasos 3 a 12 de forma separada para los bordes de la izquierda y para los bordes de la derecha.
3. Inicializar "contador" a cero y "N" (el número de iteraciones a ejecutar) a 1.
4. **Mientras** (N>contador)
5. Seleccionar aleatoriamente dos filas.
6. Calcular los parámetros de la línea "rho" y "phi" que pasen por los puntos seleccionados.
7. Para el resto de los puntos, calcular la distancia ortogonal a la línea calculada.
8. Clasificar, los puntos Inliers como aquellos cuya distancia sea menor que un cierto umbral.
9. **Si** el número de Inliers **es superior** al número de inliers encontrado hasta el momento, **entonces** guardar la solución actual y recalcular el número de iteraciones que restan **N=log(p)/log(1-porcion de inliers^2)**, siendo p=0.01 la probabilidad esperada tener un espúreo no detectado.
10. Incrementar "contador" en una unidad y en caso de que se hayan alcanzado el máximo número de iteraciones permitidas, salir con fallo.
11. Fin Repetición.
12. Calcular los parámetros de la línea con Regresión Ortogonal sólo con los puntos Inliers.
13. Calcular la línea intermedia.

Algoritmo 1. Algoritmo utilizado para la obtención de los parámetros característicos que definen la recta detectada.

El primer paso corresponde al umbralizado, en el que se buscan los posibles puntos que pertenecen a la recta a detectar. Para ello, se realiza una búsqueda de cambios de contraste por cada fila. Se trata de un sencillo proceso en el que busca tanto por la izquierda de la imagen como por la derecha, buscando el primer píxel que genera una diferencia relevante en cuanto a su intensidad en escala de grises. Ya que este procedimiento es sencillo y proclive a detectar una gran cantidad de pixeles espúreos, los pasos del 3 al 12 corresponden al algoritmo RANSAC [20], [21], [22], para el filtrado de espúreos. Como resultado del algoritmo se obtiene un subconjunto de puntos que se ajusta al modelo de una recta (inliers). Una vez seleccionados los inliers el algoritmo finaliza calculando la recta que mejor se ajusta a estos puntos. La recta está definida por una según el modelo ángulo-distancia, tal y como se muestra en la Figura 2:

$$\rho = x\cos\phi + y\sin\phi \tag{1}$$

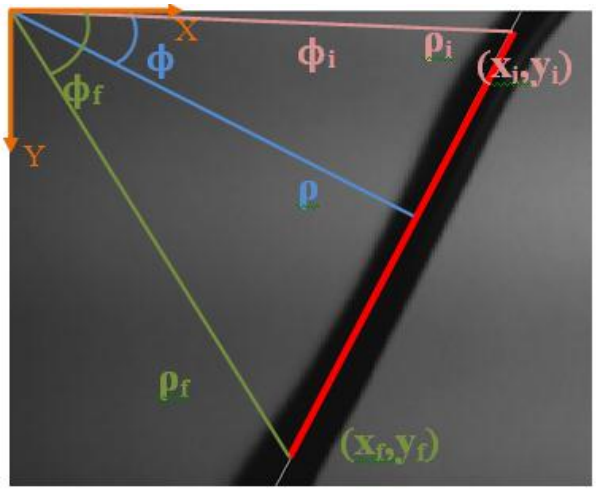


Figura 2. Representación de la recta en el espacio de distancia-ángulo.

siendo (x_i,y_i) y (x_f,y_f) las coordenadas cartesianas respecto al sistema de referencia de la imagen del punto inicial y final, respectivamente; ρ_i distancia respecto al sistema de referencia de la cámara del punto inicial; ρ_f distancia respecto al sistema de referencia de la cámara del punto final; Φ_i es el ángulo respecto al eje x del punto inicial; Φ_f es el ángulo respecto al eje x del punto final; y ρ y Φ son los parámetros que definen la recta.

Por tanto, la recta que mejor se ajusta a todos los inliers es aquella que minimiza el siguiente índice:

$$J = \frac{1}{2}\sum_{k=0}^{N-1}(x_k \cdot \cos(\phi) + y_k \cdot \sin(\phi) - \rho)^2 \quad (2)$$

siendo *N* el número total de puntos, y cuya solución es bien conocida [23], siendo $\hat{\rho} = \bar{x} \cdot cos\varphi + \bar{y} \cdot sin\varphi$, $\hat{\varphi} = \frac{1}{2} \cdot arctan\left(\frac{-2 \cdot \sigma_{xy}^2}{\sigma_y^2 - \sigma_x^2}\right)$.

A partir de los parámetros característicos de la recta se pueden recalcular los puntos extremos teniendo en cuenta la siguiente expresión:

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix} \cdot \begin{bmatrix} \rho \\ \rho \cdot \tan(\phi_i - \phi) \end{bmatrix}, \quad \begin{bmatrix} x_f \\ y_f \end{bmatrix} = \begin{bmatrix} \cos(\phi) & -\sin(\phi) \\ \sin(\phi) & \cos(\phi) \end{bmatrix} \cdot \begin{bmatrix} \rho \\ \rho \cdot \tan(\phi_f - \phi) \end{bmatrix} \quad (3)$$

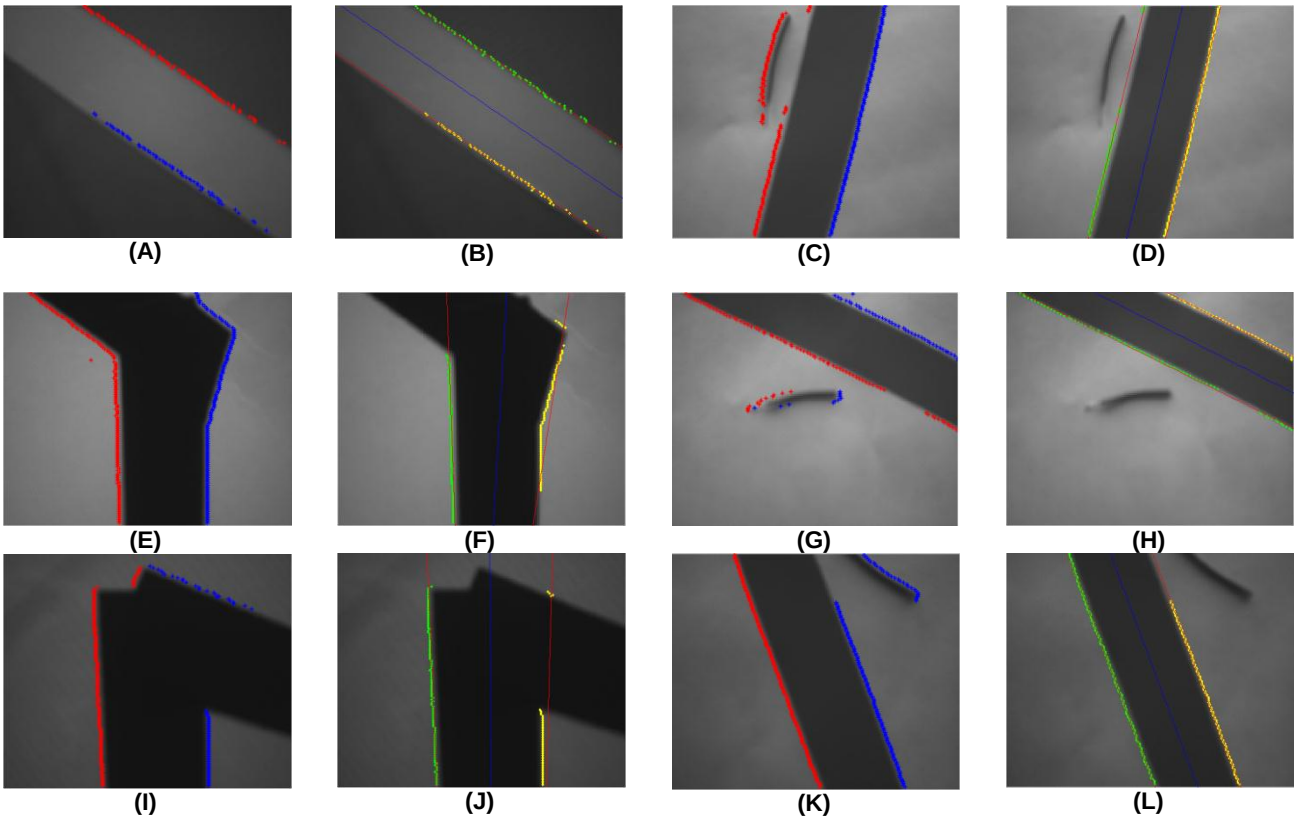


Figura 3. Representación de la recta en el espacio de distancia-ángulo.

El algoritmo ha sido probado ante diferentes condiciones con objeto de comprobar su robustez y validar la implementación. En concreto, se ha utilizado para detectar líneas de distintos colores, con mayor y menor contraste y con objetos introducidos a modo de perturbación. En la Figura 3(A) se muestra una caso en el que la línea es clara frente a un fondo oscuro, en el que se observa que funciona correctamente, tal y como se muestra en la Figura 3(B). En las Figura 3(C), (G), (K) se prueba el algoritmo con imágenes contaminadas por objetos que no pertenecen a la línea. Se observa que el RANSAC, ha sido capaz de filtrar los puntos espúreos, dando lugar a una correcta detección de la recta (ver Figura 3(D), (H), (L)). Por último, se ha estudiado el efecto de procesar una curva o rectas a tramos (ver Figura 3(E), (I)). Se observa que en esta ocasión han sido filtrados gran parte de los puntos espúreos, obteniendo de nuevo un buen ajuste de la recta.

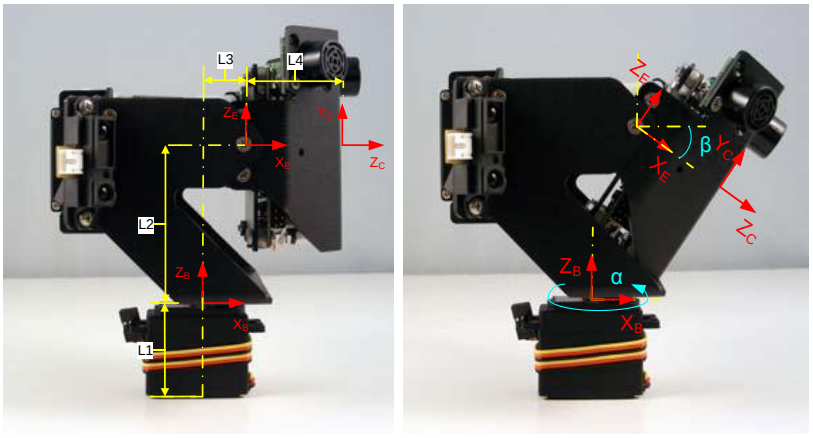
Los tiempos de respuesta del procesado y cálculo de los parámetros de envío no superan, en el peor de los casos, el tiempo de 75 milisegundos, con más de 100 pruebas realizadas ante diferentes condiciones. En el 90% de los casos el algoritmo obtiene una solución en menos de 60 milisegundos, y solo en un 5% el algoritmo no ha encontrado solución, ya sea porque el número de puntos detectados eran insuficientes o bien porque la línea estaba prácticamente horizontal.

IV. PROYECCIÓN DE LOS PUNTOS DE LA LÍNEA

Una vez obtenido los puntos extremos de la línea definidos con respecto de la imagen, se pretende obtener dichos puntos proyectados sobre el plano Z=0, que es el plano sobre el que está pintada la línea que debe de seguir el vehículo.

Para la calibración de la cámara, se ha utilizado el Calibration Toolbox de Matlab [24], obteniendo tanto los parámetros intrínsecos como los extrínsecos, esto es la transformación homogénea capaz de expresar las coordenadas de los puntos en el sistema de coordenadas de la cámara a puntos referidos al sistema de coordenadas de la imagen. Despreciando posibles aberraciones de la óptica, la matriz de parámetros intrínsecos, según el modelo pin-hole, es $\boldsymbol{K} = \begin{bmatrix} A & 0 & U_0 \\ 0 & B & V_0 \\ 0 & 0 & 1 \end{bmatrix}$, donde $A = \frac{S_U \cdot f}{CCD_U}$ y $B = \frac{S_V \cdot f}{CCD_V}$, siendo S_U y S_V el tamaño de la imagen, CCD_U y CCD_V el tamaño del CCD (Charge-Coupled Device) y f la distancia focal.

Una vez conocidos los parámetros intrínsecos es necesario obtener los parámetros extrínsecos, es decir, los que permiten referir el sistema de coordenadas del centro focal con respecto a la base de la cámara. Para determinar las transformaciones homogéneas necesarias hay que definir los sistemas de coordenadas que intervienen en la transformación. A continuación se muestran 2 figuras en las que se puede observar cómo se distribuyen los SC, así como los parámetros que se van a considerar:



(A) **(B)**

Figura 4. Sistemas de coordenadas de la torreta de la cámara.

En la Figura 4(A) se muestra de forma detallada el sistema de coordenadas de la base que corresponde con el giro vertical del primero de los dos servomotores que controlan la orientación de la cámara. Además hay otro sistema de coordenadas situado en el segundo eje de giro y finalmente otro situado en el foco de la cámara. En la Figura 4(B) aparecen los parámetros de giro (α y β), que en este caso son variables en función de las posiciones de los servos.

A continuación se muestran las distintas matrices de transformación homogéneas (MH) utilizadas:

$${}^{W}\boldsymbol{T}_{B} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$, Desplazamiento en Z $\qquad {}^{W}\boldsymbol{R}_{B} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, Giro de α en Z (4)

Las siguientes matrices permiten referir el centro focal de la cámara con respecto a la base de la torreta:

$${}^{B}\boldsymbol{T}_{E} = \begin{bmatrix} 1 & 0 & 0 & L3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & L2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$, Desplazamientos en X y Z, $\qquad {}^{E}\boldsymbol{T}_{C} = \begin{bmatrix} 1 & 0 & 0 & L4 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, Desplazamiento en X (5)

$${}^{E}\boldsymbol{R}_{1,C} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$, Giro de π/2 en Z, $\qquad {}^{E}\boldsymbol{R}_{2,C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\beta) & -\sin(\beta) & 0 \\ 0 & \sin(\beta) & -\cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, Giro de β en X (6)

Hay que destacar que en todo momento se están aplicando transformaciones respecto al sistema de coordenadas local y por lo tanto cada nueva matriz de transformación que se aplica debe de post-multiplicar al conjunto de matrices que se tiene hasta ese momento. La MH final que modela los parámetros extrínsecos de la cámara es la siguiente:

$$\boldsymbol{H}(\alpha,\beta) = {}^{W}\boldsymbol{H}_{C} = {}^{W}\boldsymbol{T}_{B} \cdot {}^{W}\boldsymbol{R}_{B} \cdot {}^{B}\boldsymbol{T}_{E} \cdot {}^{E}\boldsymbol{T}_{C} \cdot {}^{E}\boldsymbol{R}_{1.C} \cdot {}^{E}\boldsymbol{R}_{2,C} \quad (7)$$

Por tanto, la ecuación que modela la MH directa para obtener las coordenadas de los píxeles a partir de los puntos del espacio es:

$$\boldsymbol{P}_{2D} = \boldsymbol{K} \cdot \boldsymbol{H}(\alpha, \beta) \cdot \boldsymbol{P}_{3D} \quad (8)$$

Siendo $\boldsymbol{P}_{2D} \in \mathbb{R}^3$ el punto homogéneo respecto al sistema de coordenadas de la imagen, $\boldsymbol{P}_{3D} \in \mathbb{R}^4$el punto homogéneo en el espacio 3D referido al sistema de referencia global o en este caso al SC del robot, $\boldsymbol{K} \in \mathbb{R}^{3\times3}$la matriz de parámetros intrínsecos y $\boldsymbol{H} \in \mathbb{R}^{4\times4}$ la matriz homogénea de los parámetros extrínsecos de la cámara.

El modelo de proyección inversa que permite obtener puntos sobre el plano Z=0, requiere eliminar la cuarta fila y la tercera columna de $\boldsymbol{H}$. Obteniendo, por tanto, el siguiente modelo de proyección inversa:

$$\widetilde{\boldsymbol{P}}_{3D} = \boldsymbol{K}^{-1} \cdot \widetilde{\boldsymbol{H}}(\alpha, \beta)^{-1} \cdot \boldsymbol{P}_{2D} \quad (9)$$

siendo

$$\widetilde{\boldsymbol{H}}(\alpha, \beta) = \begin{bmatrix} -\sin(\alpha) & -\cos(\alpha)\cdot\cos(\beta) & (L3+L4)\cdot cos(\alpha) \\ \cos(\alpha) & -\sin(\alpha)\cdot\cos(\beta) & (L3+L4)\cdot sin(\alpha) \\ 0 & \sin(\beta) & L1+L2 \end{bmatrix} \quad (10)$$

y $\widetilde{\boldsymbol{P}}_{3D} = [X\ Y\ S]^T \in \mathbb{R}^3$ el punto homogéneo en el plano Z=0. Por tanto, para obtener las coordenadas que realmente se proyectan se debe normalizar el resultado obtenido, esto es, $\widetilde{\boldsymbol{P}}_{3D\,\mathrm{P}} = [X_F\ Y_F\ 1]^T = \left[\frac{X}{S}\ \frac{Y}{S}\ 1\right]^T$.

V. TRACKING DE LA LÍNEA

Una vez realizada la detección de la línea y la proyección de los puntos, se realiza el seguimiento de la línea con la plataforma PTU, con objeto de tener la línea dentro del campo de visión (FOV), independientemente de la orientación del vehículo. En este sentido, se propone un sencillo algoritmo para el cálculo de α que se aplica al servo (guiñada).

A partir de los puntos que definen el segmento de línea $\widetilde{\boldsymbol{P}}_i$ y $\widetilde{\boldsymbol{P}}_f$, obtenemos la distancia de la base del robot a la línea $\tilde{\rho}$ y el ángulo de la misma $\tilde{\phi}$:

$$\tilde{\phi} = arctan\left(\frac{\Delta\tilde{y}}{\Delta\tilde{x}}\right), \qquad \tilde{\rho} = \cos(\tilde{\phi})\cdot\tilde{x} + \sin(\tilde{\phi})\cdot\tilde{y} \quad (11)$$

donde, $\Delta\tilde{x} = \tilde{x}_i - \tilde{x}_f$ y $\Delta\tilde{y} = \tilde{y}_i - \tilde{y}_f$, siendo $\widetilde{\boldsymbol{P}}_i = [\tilde{x}_i\ \tilde{y}_i]^T$ y $\widetilde{\boldsymbol{P}}_f = [\tilde{x}_f\ \tilde{y}_f]^T$.

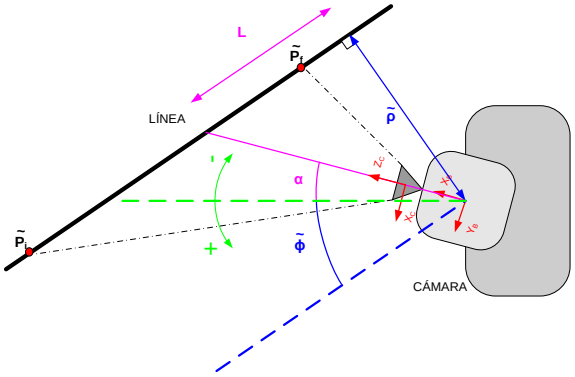


Figura 5 Proceso de detección y seguimientos de línea.

La ley de control para el seguimiento de la línea está compuesta por dos términos de corrección: uno relacionado con la propia orientación de la línea y el otro relacionado con la distancia de separación de la línea. Para el primer caso, se busca que la cámara tenga la misma orientación que la línea, mientras que para el segundo caso, se busca que la cámara apunte hacia un punto de la línea a una distancia L de la proyección de la cámara sobre la recta:

$$\alpha = -\arctan\left(\frac{\tilde{\rho}}{L}\right) + \tilde{\phi} \quad (12)$$

siendo α el ángulo que se aplica al servo.

Como se observa en la Figura 6 (A) se realiza un primer experimento en el que se mantiene ρ ≈ 0, es decir, la línea pasa en todo momento por el origen del sistema de coordenadas de la cámara. En este caso, el ángulo girado es exactamente el que se detecta en la imagen (ambas respuestas están superpuestas). En la Figura 6 (B), se realiza un segundo experimento, en que se fija el valor del ángulo de la línea para que en todo momento sea Φ ≈ 0. En este caso, se pretende mostrar que, aunque el ángulo sea siempre nulo, la cámara es capaz de apuntar hacia la línea cuando ésta se desplaza horizontalmente. Se observa que desplazamientos positivos en ρ (la línea está a la derecha de la cámara) generan ángulos negativos (la cámara gira en el sentido de las agujas del reloj).

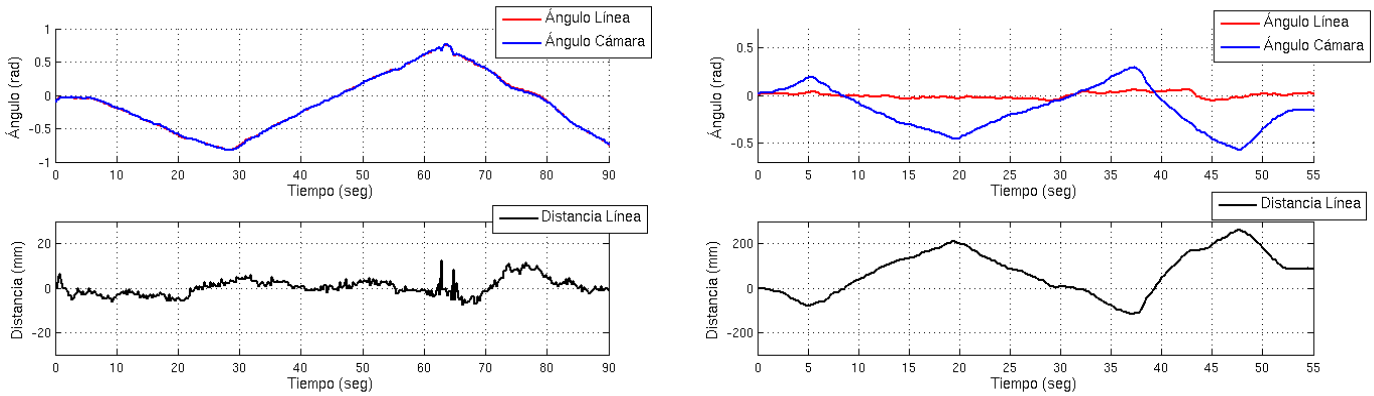


Figura 6. Seguimiento de línea con $\tilde{\rho} = 0$ (A) y $\tilde{\phi} = 0$ (B).

Finalmente, se ha realizado un tercer experimento en el que se ha generado un movimiento arbitrario entre la línea y la cámara. Se observa que en términos generales la cámara adopta el ángulo que detecta de la línea, en cuyo caso las discrepancias vienen motivadas por la separación de la línea con respecto al origen de la cámara, que no es nula para este experimento.

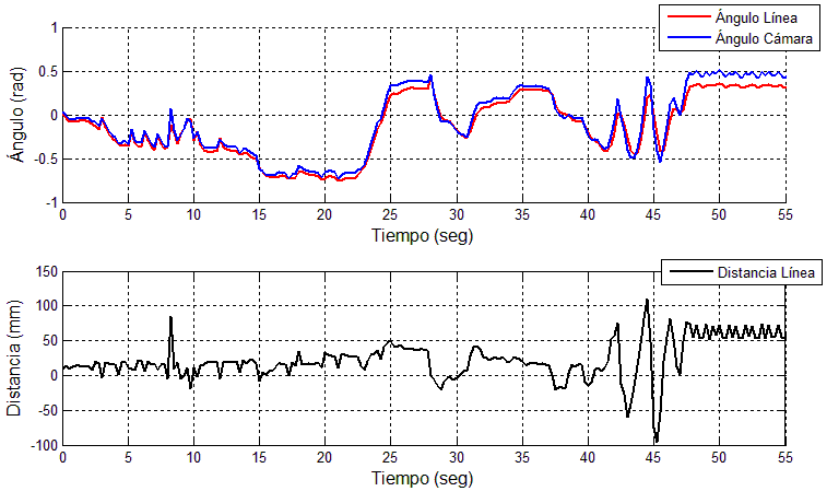


Figura 7. Seguimiento de línea para valores arbitrarios de $\tilde{\rho}$ y $\tilde{\phi}$.

VI. CONCLUSIONES Y TRABAJOS FUTUROS

En este artículo se ha presentado resultados intermedios del Proyecto de Investigación PISALA, cuyo objetivo es el seguimiento de líneas por visión artificial con AGVs. En concreto, en el artículo se describe la arquitectura hardware utilizada para este propósito, siendo el sistema de visión embebido CMUCAM3 montado sobre una PTU el elemento más relevante de la aplicación descrita. Teniendo en cuenta las características de la CMUCMA3 y las limitaciones que presenta, se ha propuesto un algoritmo de detección de líneas que ha demostrado ser robusto frente a diversas condiciones, tales como cambios de iluminación, formas de la línea y datos espúreos en la imagen. Finalmente, el artículo describe un sencillo algoritmo para el seguimiento de la línea utilizando la PTU, en la que se ha demostrado su correcto funcionamiento ante un conjunto de movimientos arbitrarios.

Como trabajos futuros, se pretende seguir avanzando en el proceso de automatización de manera que el vehículo converja a la línea detectada, evitando posibles objetos del entorno.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente subvencionado con los Proyectos de Invetigación GVPRE/2008/034, PAID-06-08-3246 (20081144) y DPI2001-2689-C03-02.

REFERENCIAS

[1] C. Lafuente, “AGV: automatizar el transporte y los flujos internos: flexibilidad, precisión y seguridad,” Manutención y almacenaje, n. 42, pp. 71–77, 2005.

[2] www.egemin.com.

[3] www.robocoaster.com.

[4] M.B. Duinkerken, J.A. Ottjes, G. Lodewijks, “Comparison of routing strategies for AGV systems using simulation”, Proceedings of Winter Simulation, pp. 1523-1530, 2006.

[5] S. Thrun, W. Burgard, D. Fox, “Probabilistic Robotics”, MIT Press, 2005.

[6] K.R.S. Kodagoda, W.S. Wijesoma and E.K. Teoh, “Fuzzy speed and steering control of an AGV”, The IEEE transactions on control systems technology , Vol.10, No.1, pp112-120, 2002.

[7] W.S. Wijesoma, K.R.S. Kodagoda and E.K. Teoh, “Stable Fuzzy State Space Controller for Lateral Control of an AGV”, Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology , Vol. 32, pp189-201, 2002.

[8] M. Brady, H. Durrant-Whyte, H. Hu; J. Leonard, P. Probert;, B.S.Y. Rao, “Sensor-based control of AGVs”, vol. 1, nº 2, pp. 64-70, 1990.

[9] G. Garibotto, S. Masciangelo, P. Bassino, C. Coelho, A. Pavan, and M. Marson, “Industrial exploitation of computer vision in logistic automation: autonomous control of an intelligent forklift truck,” in Proc. Int. Conf, on Robotics and Automation, vol. 2, 1998, pp. 1459 –1464.

[10] J. Pagés, X. Armangué, J. Salvi, J. Freixenet, and J. Martí, “A computer vision system for autonomous forklift vehicles in industrial environments,” in 9th. Mediterranean Conf. on Control and Automation, 2001, pp. 379–384.

[11] M. S. J. Yoder, “Automatic pallet engagment by a vision guided forklift,” in IEEE Conference on Robotics and Automation, 2005.

[12] M. Mora, V. Suesta, L. Armesto, and J. Tornero, “Factory management and transport automation,” in IEEE Conference on Emerging Technologies and Factory Automation, vol. 2, 2003, pp. 508–515.

[13] L. Armesto, M. Mora, and J. Tornero, “Supervisión, teleoperación y navegación de vehículos industriales y su integración en el sistema de gestión,” Revista Iberoamericana de Automática e Informática Industrial, vol. 2, pp. 55–63, 2005.

[14] L. Armesto, J. Tornero, AutoTrans: Management and transport automation in warehouses”, Industrial Simulation Conference, vol. 1, pp. 236-241, 2005.

[15] Li. W;Xu C.;Xiad, Q; and Xu, X. “Visual Navigation of an autonomous robot using white line recognition”, Proceedings of the 2003 IEEE Iternational Conference on Robotics and Automation (ICRA), 2003.

[16] T.H. Hong, T. Chang, C. Rasmussen and M. Shneier, “Road Detection and Tracking for Autonomous Mobile Robots”, Proceedings of SPIE Aerosense Conference, Vol. 4715, 2002.

[17] M. Ollis and A. Stentz, “Vision based perception for an automated harverster”, In proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS), vol. 2, pages 1838-1844, 1997.

[18] B. Ross, J. Bares and C. Fromme “A semi-autonomous robot for stripping paint from large vessels”, The International Journal of Robotics Research, 22 (7-8), pp. 617-626, 2003.

[19] www.cmucam.org.

[20] M. Fischler and R. Bolles, “Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography,” Comm. ACM, vol. 24, no. 6, pp. 381- 395, June 1981.

[21] J. Matas and O. Chum, “Randomized RANSAC with Td;d Test,” Image and Vision Computing, vol. 22, no. 10, pp. 837-842, Sept. 2004.

[22] “Randomized RANSAC with Sequential Probability Ratio Test,” Proc. Int’l Conf. Computer Vision, vol. 2, pp. 1727-1732, Oct. 2005.

[23] G. Araujo and M. Aldon, “Line extraction in 2d range images for mobile robotics,” Journal of Intelligent and Robotic Systems, no. 40, pp. 267–297, 2004.

[24] www.vision.caltech.edu/bouguetj/calib_doc.