UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MORELOS

Facultad de Ciencias

Navegación de un agente autónomo usando visión estereoscópica

T E S I S

Que para obtener el titulo de:

Licenciado en Ciencias
(Ciencias Computacionales)

Presenta:

Esaú Eliezer Escobar Juárez

Director de tesis: Dr. Bruno Lara Guzmán

Cuernavaca, Morelos, México. 20 de marzo de 2011

Agradecimientos

Gracias a Dios, por acompañarme en cada momento y por fortalecer mi voluntad e iluminar mi mente durante todo el periodo de estudio.

A mi esposa Clara le doy gracias infinitamente por su apoyo, comprensión y confianza que me han ayudado enormemente en estos años que llevamos caminando juntos. Debo decirte que soy completamente feliz por tenerte a mi lado y que tú junto con nuestros hijos son lo más importante en mi vida.

Gracias a mi familia en especial a mi Mamá Pilar por ser mi apoyo y por darme tu amor. Gracias porque tú me has enseñado con el ejemplo que las cosas se obtienen con esfuerzo y dedicación. También gracias a mis hermanos Dulce y Saulo por sus buenos consejos.

Gracias a mi tío Alberto pues me brindo todo su apoyo para continuar con mis estudios.

Agradezco también de manera especial al profesor Dr. Bruno Lara por transmitirme sus conocimientos, por su paciencia, por guiarme y por aconsejarme para lograr concluir este trabajo de tesis y mi licenciatura.

Por último agradezco a todos los maestros y amigos que he conocido en la licenciatura, pues desde que llegue a la universidad me dieron ánimo y su amistad.

Índice general

Índice de figuras

Índice de cuadros

Resumen

El objetivo del trabajo reportado en esta tesis es entrenar a un agente autónomo para navegar en un ambiente haciendo uso de visión estereoscópica. El agente es un robot móvil que cuenta con un par estereoscópico montado en su parte superior. Dicho agente obtiene un mapa de disparidad a través del par estereoscópico, este mapa provee al agente con diferentes regiones correspondientes a diferentes distancias. Las diferentes regiones del mapa de disparidad están codificadas con diferentes intensidades. Utilizando los valores de los sonares el agente deberá ser capaz de hacer una asociación entre regiones de intensidad y distancia a objetos. Se propone obtener esta asociación mediante el uso de un modelo directo. Un modelo directo es un modelo interno que provee al agente de las predicciones de situaciones sensoriales basadas en comandos motrices. El modelo directo se codificará haciendo uso de una red neuronal artificial.

Capítulo 1

Introducción

El desarrollo de robots que puedan llevar a cabo tareas confiablemente de manera autónoma ha sido durante varias décadas un tema de gran interés para la comunidad científica. La creciente evolución de los sistemas computacionales y su aplicación a las diversas áreas del conocimiento es ahora lo que nos permite pensar en crear artefactos autónomos y que además muestren un cierto grado de inteligencia.

En este documento consideraremos el problema de entrenar un agente para que tenga la capacidad de extraer la información de su sistema sensorial, lo cual le permita desenvolverse en un ambiente de manera autónoma. Los robots autónomos son entidades físicas con capacidades de percepción sobre un entorno y que actúan sobre el mismo en base a dichas percepciones, sin supervisión directa de otros agentes naturales o artificiales.

En este trabajo se planea que el sistema sensorial del agente este compuesto de una cámara de visión estéreo y un anillo de sonares.

En la mayoría de los trabajos relacionados con sistemas robóticos y visión estéreo no se utiliza un modelo cognitivo para que el robot aprenda a extraer la información del entorno para dirigir sus acciones, sino que se utilizan modelos para construir mapas y hacer planeación de rutas. Llamamos modelos cognitivos a aquellos modelos que están inspirados en estudios de la cognición humana.

Este trabajo está desarrollado en el contexto de la cognición embebida que forma parte de la nueva visión de la inteligencia artificial [2]. La cognición embebida declara como punto central que los agentes deben tener un cuerpo e interactuar con su entorno, puesto que es a través de esta interacción que se aprende y se entiende este.

1.1. Nueva Inteligencia Artificial

En los años en que la inteligencia artificial comenzó a surgir se consideraba que los grandes problemas que se resolverían con las técnicas de inteligencia artificial eran los relativos a la demostración de teoremas matemáticos, la comprensión del lenguaje natural y otras tareas formales. De ahí que se pensara en la inteligencia artificial como una tarea de resolución simbólica, es decir la resolución de problemas mediante la manipulación y procesamiento de símbolos, que no son otra cosa que representaciones abstractas del mundo.

Se pensó para estos fines en el paradigma jerárquico lineal de procesamiento de información denominado también como paradigma Percepción, Planificación, Acción o PPA [3], la denominación de jerárquico se debe a que cada operación se lleva a cabo secuencialmente, en dicho orden y porque cada operación toma como entrada las salidas de la operación anterior de manera unidireccional.

La fase de **percepción** es clave en todo robot autónomo, ya que para poder influir sobre el entorno, es necesario que exista una manera de percibir el estado actual del mismo.

En la fase de **planificación** el robot decide qué acciones llevar a cabo tomando como base las percepciones recién adquiridas y sus propios objetivos. Para poder efectuar la planificación, las percepciones deberán haber sido preprocesadas y transformadas en una abstracción del entorno.

En la fase de **acción** se llevan a cabo las acciones de alto nivel que se seleccionaron en la fase de planificación, traduciéndolas en movimientos de los actuadores del robot.

En la Figura 1.1 podemos ver cómo se aplican las ideas de la inteligencia artificial clásica a los robots, en las cuales el investigador trata de abstraer la información del entorno, crea un programa que lidie con esas representaciones simbólicas, el cual posteriormente se le carga al robot. Como siguiente paso el robot debe, con su sistema sensorial, abstraer también la información simbólica del mundo para operar con el programa cargado en memoria y así poder realizar sus tareas [3].

Posteriormente, los investigadores comenzaron a darse cuenta de que las tareas realmente complejas son las relacionadas con la abstracción del mundo (percepción), las cuales han resultado ser computacionalmente intratables, y su transformación a símbolos no ha sido resuelta todavía. Para poder efectuar la planificación de la manera planteada, era necesario que en la fase de percepción pudiera llevarse a cabo una complicada tarea de abstracción para proveer a la fase de planificación con los símbolos de alto nivel que ésta

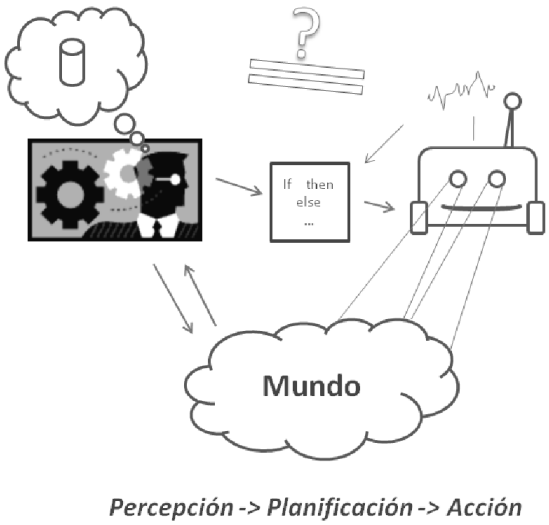


Figura 1.1: Inteligencia artificial clásica.

requería para operar.

Por otra parte, esta visión de la inteligencia artificial considera a los procesos de cognición como módulos que reciben, modifican y pasan la información disponible en el entorno y provee la oportunidad de considerar y trabajar en módulos aislados con habilidades especializadas. Sin embargo de ahí que surjan otros problemas. Primeramente, algunas conductas se desarrollan y se aprenden mejor cuando un agente interactúa dinámicamente con su entorno. Segundo, la complejidad de interrelación de los diferentes módulos aumenta considerablemente cuando se incrementan el número de éstos, debido a la necesidad de programar módulos para conductas específicas, además de que siempre habrá casos no considerados [4]. Tercero, estos agentes caen en el problema de cimentación, que implica que el agente es ajeno al entorno ya que no aprendió a través de la interacción con él [5].

Es a raíz de estos problemas que la inteligencia artificial comenzó a tomar otro rumbo. Inspirado en la ciencias cognitivas se crea el paradigma Percepción- Acción o PA en el cual se deja de poner énfasis en la abstracción de símbolos y se comienza a pensar en las acciones ligadas a las percepciones. Con este enfoque los investigadores comienzan a darse cuenta que la inteligencia no está limitada al razonamiento abstracto, estas ideas están plasmadas en el artículo de Rodney Brooks "Los elefantes no juegan ajedrez" [6].

En dicho artículo Brooks expone que un animal no necesita llevar a cabo razona-

mientos simbólicos de alto nivel para sobrevivir en su entorno. Lo realmente imprescindible es que el animal disponga de una manera de percibir su entorno y actuar sobre el mismo reactivamente, sin llevar a cabo un razonamiento intensivo.

El paradigma PA presenta requerimientos mucho menores para las tareas de percepción, ya que al no haber planificación no resulta necesario abstraer lo percibido transformándolo en símbolos. En lugar de ello, las percepciones son pasadas al módulo de acción con un mínimo de preprocesamiento.

En un nuevo enfoque con respecto al tema de percepción ahora se considera la entrada de datos sensoriales y la acción motriz (salida de datos) como parte del mismo proceso cognitivo. Este punto de vista hace hincapié en el papel que juegan los estados internos, como son los objetivos o metas, despreciando a diferentes niveles las condiciones sensoriales externas [4].

Mucho más aun, recientemente se ha valorado la idea de que la anticipación de acciones y los estados sensoriales pueden influenciar la conducta. Se considera que la anticipación juega un papel importante en la coordinación, planeación y realización de la conducta. El enfoque del proceso lineal de información a cambiado a otro en el cual el flujo de información ya no es una trayectoria en una sola dirección [4].

En esta misma corriente de pensamiento, las representaciones sensoriales también se consideran como consecuencias de acciones. Cualquier acción realizada por un agente sobre su entorno tiene efectos sensoriales y son la razón principal para la conducta. La planeación y el control de acciones se vuelven anticipatorios cuando están dirigidos por las situaciones sensoriales deseadas o los efectos deseados de las acciones.

En este punto podemos analizar en la Figura 1.2 como se realiza el procesamiento de las percepciones sensoriales y las acciones para que emerja el comportamiento como un mecanismo que el mismo agente aprende a descubrir y a relacionar por medio de la interacción con el mundo. Como se muestra, no se lleva a cabo una abstracción simbólica, ni un flujo de información unidireccional sino que es más bien una interacción entre la percepción y la acción.

1.2. Planteamiento del problema

La tarea de obtener la información en 3D de un sistema de percepción visual estéreo ha sido ampliamente investigada y se han desarrollado diversos métodos para obtener

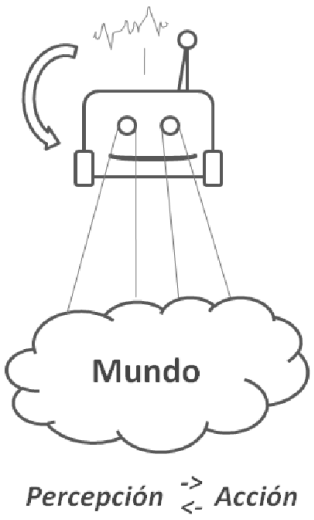


Figura 1.2: Nueva inteligencia artificial.

las correspondencias necesarias en las imágenes (*sección 2.1.5*) que permitan computar la profundidad de los distintos objetos que se encuentran situados en la escena capturada.

Sin embargo, cuando se ha requerido utilizar esta información para que un robot móvil pueda navegar en un ambiente lo que se ha hecho es tratar de adaptar la información sensorial simbólica (en cms) a una representación que el robot pueda manejar y lo habilite para llevar a cabo sus tareas ([7], [8], [9]).

Nos podemos dar cuenta de que la información obtenida del sistema de percepción estéreo esta en términos ajenos a las características intrínsecas del robot. Para aclarar esto podemos pensar en qué significa para el robot que el sistema de visión le reporte que hay un objeto a 50 centímetros de distancia. Por ejemplo en función del tamaño del robot esta distancia puede significar distintas posibilidades. En el caso de un robot con un tamaño de 5 cm. y sistema motriz de reducidas dimensiones un objeto a esta distancia no supone una colisión inmediata, puesto que es muy posible que dicha distancia represente la ejecución de una larga secuencia de movimientos, por lo cual si la tarea es evadir obstáculos entonces avanzar puede ser una acción adecuada. En el caso de un robot de 50 cm. con un sistema motriz de grandes dimensiones el objeto es muy probable que represente una colisión casi inmediata por lo que girar puede ser la acción más adecuada si la tarea es evadir obstáculos.

En el contexto de la nueva inteligencia artificial surge la pregunta ¿cómo lograr

dotar a un robot de la capacidad de aprender a relacionar su percepción con un significado intrínseco a sus capacidades motoras y a su propio cuerpo? (Figura 1.3).

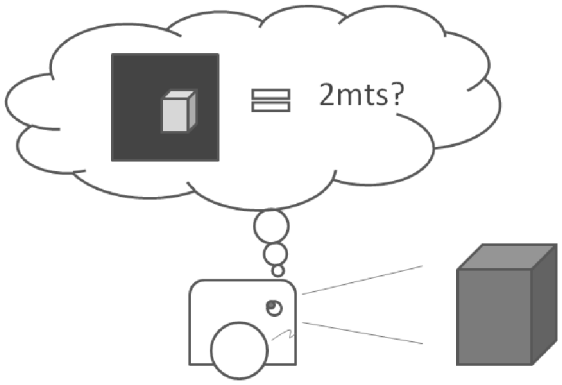


Figura 1.3: ¿Cómo lograr que el propio robot haga la asociación de la información sensorial con su propia noción de distancia?.

El objetivo principal del trabajo propuesto es el de dotar a un agente con una noción de distancia a los objetos que lo rodean. Esta noción deberá estar relacionada a la arquitectura propia del robot, a sus características físicas y capacidades motrices.

Se propone utilizar un modelo directo para formar una representación multimodal entre la percepción visual, táctil y los comandos motrices.

La hipótesis a considerar es: "Utilizando un modelo directo es posible dotar a un agente de una noción de distancia que corresponda a sus características fisicas"

En términos más específicos se propone utilizar una cámara estéreo para generar el mapa de disparidad, posteriormente entrenar al robot para que extraiga la información de distancia y prediga una colisión a corto plazo a partir de dicho mapa, con lo cual pueda navegar en el entorno.

1.3. Estado del Arte

Se presenta una revisión de algunos trabajos relacionados con el problema propuesto lo cual ayudara a identificar puntos importantes a tener en cuenta al desarrollar la metodología de solución.

Navegación

Se conoce como navegación a la metodología que permite guiar el curso de un

robot móvil a través de un entorno con obstáculos, el cual normalmente se denomina arena. Existen diversos esquemas, pero todos ellos poseen en común el afán por llevar el vehículo a un destino de forma segura. El desarrollo de sistemas de navegación en robots es un tema en el cual existen diversos trabajos de investigación.

Se ha considerado la planeación de trayectorias como uno de las principales mecanismos de la navegación. Como en el trabajo presentado en [10] que se basa en árboles aleatorios de exploración rápida. También en la navegación tiene importancia la tarea de creación del mapa del entorno del agente, como lo menciona el artículo [11] que se enfoca en la creación de mapas para ambientes dinámicos.

Uno de los comportamientos característicos en la tarea de navegación es el de evadir obstáculos y ha sido tema central en la mayoría de los trabajos de investigación, sin embargo podemos ver que también se han tomado en cuenta otros comportamientos, tal como el de búsqueda de fuentes de luz [12].

Para estos trabajos también se pensó en los sensores de rango como la principal herramienta para navegar, éstos son los sensores de ultrasonidos, infrarrojos y láser. Como en [13], en el cual se utilizan sonares con sensores infrarrojos para construir un mapa de un entorno y hacer planificación de rutas. Dentro de este ámbito se encuentra también la investigación reportada en [14], que hace uso de un sensor láser de 180 grados junto con una red de sensores láser dispuestos en puntos estratégicos en el entorno.

Visión Estéreo

Por otro lado se ha venido desarrollando el uso de visión estéreo para la navegación. Describiremos algunos artículos en los cuales se genera el mapa de disparidad con el cual se forma una malla de ocupación y también trabajos en los que la disparidad se utiliza para la detección de las coordenadas 3-D de obstáculos.

En el artículo [15] se examina el uso de la región de interés. Se crea un mapa de disparidad a partir de imágenes de una cámara estereoscópica utilizando el método de correspondencia denominado suma de las diferencias absolutas (SAD) (*sección 2.1.5*). De dicho mapa se extrae solo la región central (región de interés) sobre la línea epipolar (*sección 2.1.2*) en un área de 320 x 20 píxeles de la cual se obtienen las distancias a los objetos, al final se utilizan estos datos para navegar con un algoritmo simple.

En el artículo [7] se utiliza la visión estéreo para detección de obstáculos. Se calcula el mapa de disparidad sobre las imágenes estéreo con el método de la suma de las diferencias

de los cuadrados (SSD) (*sección 2.1.5*), para posteriormente detectar los obstáculos midiendo el cambio en las intensidades (distancias) de dos píxeles en dos filas contiguas en la imagen de disparidad en la misma columna. Si este valor es mayor de un determinado umbral entonces el píxel es marcado como correspondiente a un obstáculo (Figura 1.4).

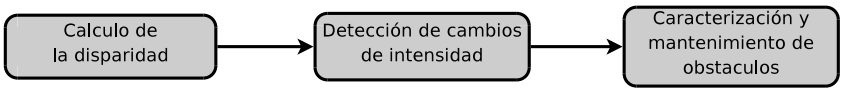


Figura 1.4: Modelo de Collins para detección de obstáculos.

En [8] se utiliza la visión estéreo para detección de peligros potenciales. Se obtiene el mapa de disparidad para formar probabilísticamente una malla 3D de ocupación y posteriormente construir un mapa local de zonas seguras (Figura 1.5).

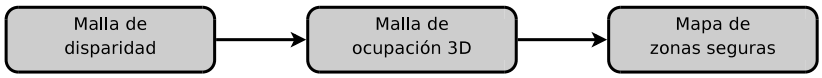


Figura 1.5: Etapas de procesamiento de Murarka para caracterización de peligros potenciales.

En [9] se usa visión estéreo en tiempo real para navegar. La metodología utilizada está conformada de las siguientes etapas: calcular un mapa de disparidad a través de un sistema de visión trinocular, con esta información construir probabilísticamente una malla de ocupación la cual se va actualizando en tiempo real, como parte final se utiliza planeación de rutas para navegar de forma segura (Figura 1.6).

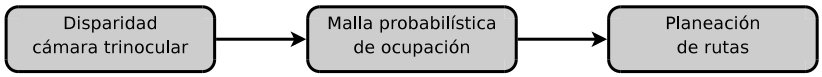


Figura 1.6: Propuesta de Murray para la navegación segura en tiempo real.

Modelos Directos

También se han desarrollado modelos cognitivos diversos, entre ellos están los modelos internos que como se menciona en [16] son mecanismos neuronales que imitan las características de entrada y salida de un agente. Los modelos internos directos (*sección 2.2*) pueden predecir las consecuencias sensoriales al tiempo siguiente (S_{t+1}) a partir de los comandos motrices (M_t) y el estado sensorial actual (S_t), como se puede apreciar en la Figura

1.7. Los modelos internos inversos, por el contrario, pueden sugerir un comando motriz (M_t) en función de la información de la trayectoria deseada, es decir en función del estado sensorial actual (S_t) y los efectos sensoriales deseados al tiempo siguiente (S_{t+1}), como se muestra en la Figura 1.8.

Figura 1.7: Diagrama general del modelo directo.

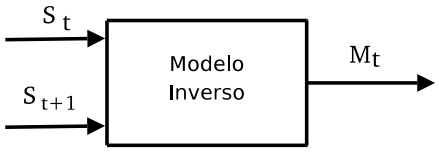


Figura 1.8: Diagrama general del modelo inverso.

En el contexto de los modelos internos encontramos la investigación presentada en [1] que plantea su uso para hacer emerger habilidades cognitivas tales como la percepción, mostrando el funcionamiento a través de un experimento con un agente simulado que aprende a distinguir corredores y caminos cerrados, sin la necesidad de la representación implícita (Figura 1.9).

En el mismo campo encontramos la investigación presentada en [4] la cual está enmarcada en la cognición embebida y utiliza un modelo directo (Figura 1.10) a fin de conseguir predecir situaciones sensoriales a corto o largo plazo, lo cual se puede aplicar en diversos campos, y como ejemplo demostrativo se implementa en un robot que sigue una fuente de luz evitando obstáculos de manera oportuna, este trabajo utiliza representaciones sensoriales multimodales como son la visión con una cámara lineal omnidireccional y sensores táctiles.

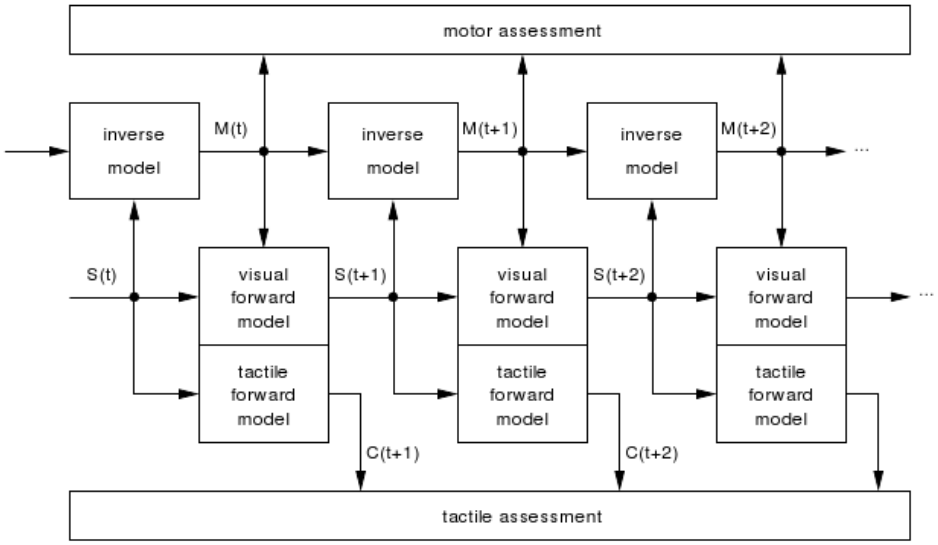


Figura 1.9: Modelo interno de R. Moller [1].

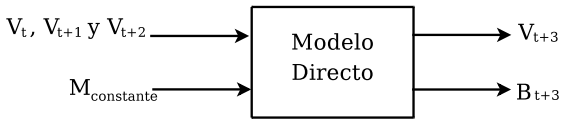


Figura 1.10: Modelo directo utilizado por B. Lara para predecir situaciones sensoriales multimodales.

Capítulo 2

Marco Teórico

En este apartado abordaremos los principales aspectos teóricos necesarios para sustentar la metodología de solución propuesta (*sección 3*) en el presente trabajo.

2.1. Visión estereoscópica

La visión estéreo usa dos cámaras separadas por una distancia conocida para obtener un mapa del terreno frente al sistema. Se basa en el mecanismo humano para percibir la profundidad en el cual el cerebro interpreta la realidad a partir de las imágenes que le proporcionan los dos ojos. Estas imágenes presentan pequeñas diferencias entre sí debidas a la separación que existe entre los ojos. La *disparidad* (diferencia entre las dos proyecciones de la posición del objeto en las retinas) entre estas imágenes es utilizada por el cerebro para percibir la profundidad, siendo la base de la denominada visión estereoscópica.

Desde un punto de vista computacional, un sistema de visión estéreo debe resolver dos tipos de problemas. El primero de ellos, conocido como *correspondencia*, consiste en determinar qué elemento en la imagen izquierda corresponde a cuál elemento en la imagen derecha; el segundo problema que un sistema estéreo debe resolver es la *reconstrucción*, que consiste en obtener la estructura en tres dimensiones de un objeto que está en el mundo, a partir de imágenes de dicho objeto.

Para poder estimar la posición en 3D de los objetos, es necesario resolver el problema de la correspondencia. Debemos tener en cuenta que debido a la variación del punto de vista, se presenta el problema de que pueden existir puntos que sólo son visibles en una de las imágenes, a este problema se le conoce como *problema de oclusión*. Por lo tanto un buen

algoritmo para resolver el problema de las correspondencias debe determinar qué áreas en cada imagen están ocultas en la otra para no emparejarlas.

La percepción en 3D del entorno se consigue gracias a la interpretación de la diferencia calculada en los planos de la imagen de cada cámara, llamada *disparidad*, entre elementos correspondientes.

2.1.1. Calibración de cámaras

Para obtener información confiable del entorno a través de un sistema de visión, es necesario conocer los parámetros que definen la proyección de un punto en la escena **X** en 3D al punto **x** en el plano de la imagen en 2D (Figura 2.1). Al proceso realizado para estimar dichos parámetros se le conoce como *calibración de cámara*.

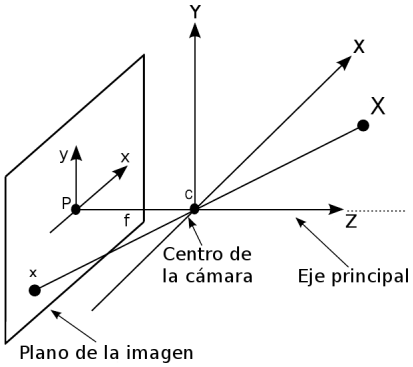


Figura 2.1: Geometría de la cámara pinhole. El centro de la cámara está localizado en el origen. El plano de la imagen está localizado atrás del centro de la cámara.

Una modificación sencilla al modelo de pinhole para entenderlo mejor es mover el plano imagen para que se encuentre entre el lente y el objeto. Esto se muestra en la Figura 2.2 [17].

En un estado ideal, las cámaras de un sistema estéreo, tienen exactamente la misma distancia focal y los ejes ópticos son paralelos. En la practica, las cámaras son imperfectas, sufren de distorsión de los lentes, su distancia focal difiere y sus ejes ópticos están desalineados. El objetivo de la calibración de las cámaras del sistema estéreo es determinar 2 conjuntos de parámetros, *intrínsecos* y *extrínsecos*, que posteriormente nos permitan compensar las imperfecciones del sistema estéreo.

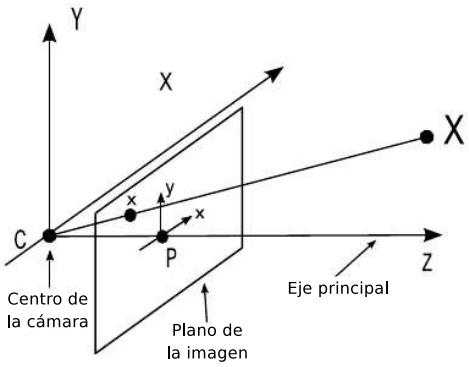


Figura 2.2: Geometría de la cámara pinhole. El centro de la cámara está localizado en el origen. El plano de la imagen está localizado enfrente del centro de la cámara.

Los parámetros intrínsecos, que determinan la generación de la imagen en la cámara, corrigen la distorsión de los lentes y la desigualdad en la distancia focal.

Los parámetros extrínsecos, que relacionan las coordenadas del mundo real con las de la cámara, determinan el desplazamiento espacial de las dos cámaras, incluyendo la linea base del sistema estéreo (*baseline*) y alguna desviación de los ejes ópticos paralelos.

Estos parámetros entonces pueden ser usados para transformar las imágenes de las cámaras a una *posición estándar*, que son, imágenes que podrían ser vistas por cámaras pinhole con ejes ópticos paralelos. A esta transformación se le conoce como *rectificación*.

La idea principal al calibrar una cámara es obtener la matriz P que contiene los parámetros de la cámara, la cual cumple con:

$$\mathbf{x} = P\mathbf{X} \tag{2.1}$$

De manera general la matriz de proyección P está definida de la forma:

$$P = KR[I| - \tilde{C}] \tag{2.2}$$

donde $\tilde{C}$ representa las coordenadas del centro de la cámara en el marco de referencia del mundo, la matriz R es la matriz de rotación (de 3×3) representando la orientación del marco de referencia de la cámara, los valores de R y $\tilde{C}$ representan los parámetros extrínsecos, y K es la matriz de calibración definida como:

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ & \alpha_y & y_0 \\ & & 1 \end{bmatrix} \tag{2.3}$$

donde x_0 y y_0 son las coordenadas del punto principal en píxeles, s el factor de torcimiento que normalmente es 0 a menos que los ejes x y y no sean perpendiculares y α_x y α_y son la distancia focal expresada en píxeles, es decir, $\alpha_x = fm_x$ y $\alpha_y = fm_y$ de donde f es la distancia focal y m_x y m_y son los factores de escala en las direcciones x y y en número de píxeles por unidad de distancia; La matriz K representa los parámetros intrínsecos.

De acuerdo a la formula (2.1) se tiene:

$$\begin{bmatrix} \rho x_i \\ \rho y_i \\ \rho \end{bmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix} \tag{2.4}$$

al resolver la parte derecha y dividir entre el último elemento obtenemos:

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}} \\ \frac{p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24}}{p_{31}X_i + p_{32}Y_i + p_{33}Z_i + p_{34}} \\ 1 \end{bmatrix} \tag{2.5}$$

despejamos las ecuaciones e igualamos a cero:

$$p_{11}X_i + p_{12}Y_i + p_{13}Z_i + p_{14} + x_i p_{31}X_i + x_i p_{32}Y_i + x_i p_{33}Z_i + x_i p_{34} = 0 \tag{2.6}$$

$$p_{21}X_i + p_{22}Y_i + p_{23}Z_i + p_{24} + y_i p_{31}X_i + y_i p_{32}Y_i + y_i p_{33}Z_i + y_i p_{34} = 0 \tag{2.7}$$

Las ecuaciones (2.6) y (2.7) se obtienen para cada i-ésimo punto de correspondencia entre $\mathbf{x}_i \leftrightarrow \mathbf{X}_i$, con lo que utilizando 6 puntos tenemos 12 ecuaciones para 11 incógnitas, ya que el valor de $p_{34} = 1$, lo cual se resuelve por medio de métodos numéricos para obtener los valores de la matriz P.

2.1.2. Geometría epipolar

El análisis de la geometría epipolar y la matriz fundamental es muy importante, debido a que permite simplificar el proceso de búsqueda de pares de puntos correspondientes.

La geometría epipolar es la geometría proyectiva intrínseca entre dos vistas. Ésta es independiente de la estructura de la escena, y sólo depende de los parámetros intrínsecos de la cámara y de la posición relativa de las cámaras.

La matriz fundamental F encapsula esta geometría intrínseca. Si un punto en el espacio X es mapeado a la primera vista como x, y como x' a la segunda vista, entonces los puntos de las imágenes satisfacen la relación $x'^T F x = 0$ [18]. Las entidades geométricas que intervienen en la geometría epipolar se pueden observar en la Figura 2.3. La terminología es:

- **Epipolo** es el punto de intersección de la línea que une los centros de las cámaras (línea base) con el plano de la imagen. Equivalentemente, el epipolo es la imagen en una vista del centro de la cámara de la otra vista.
- **Plano epipolar** es un plano que contiene la línea base. Una familia de planos epipolares se conoce como pencil.
- **Línea epipolar** es la intersección de un plano epipolar con el plano de la imagen. Todas las líneas epipolares se intersectan en el epipolo.

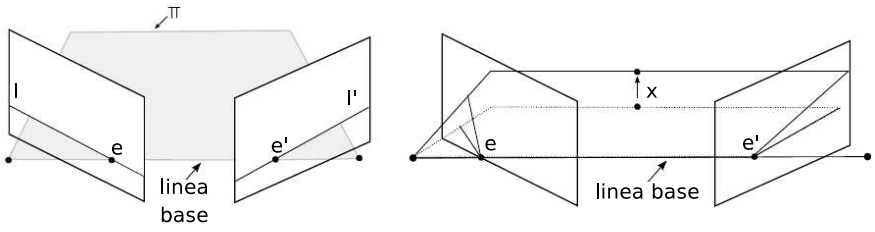


Figura 2.3: Geometría epipolar.

La geometría epipolar entre dos vistas es esencialmente la geometría de la intersección de los planos de la imagen, con el pencil de planos teniendo la línea base como eje (la línea base es la línea que une los centros de las cámaras). Usualmente se utiliza esta geometría para restringir la búsqueda de puntos correspondientes.

Suponemos un punto X en el espacio 3D que es mapeado a dos vistas, como x en la primera, y como x' en la segunda. Como podemos ver en la Figura 2.4 los puntos x y x' en las imágenes, el punto X en el espacio y los centros de las cámaras son coplanares.

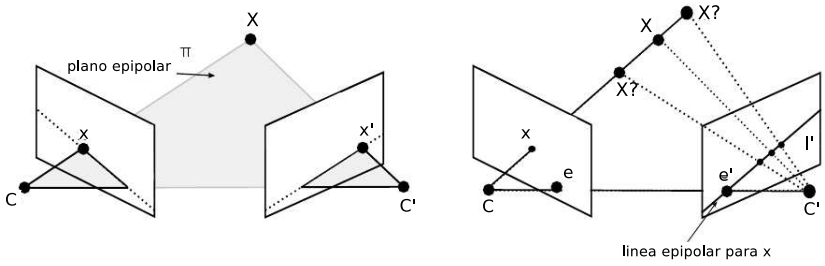


Figura 2.4: Geometría de los puntos correspondientes.

Dado un par de imágenes, como se ve en la Figura 2.4, para cada punto x en una imagen existe una línea epipolar correspondiente en la otra imagen. El punto x' en la segunda imagen que es correspondiente al punto x debe estar sobre la línea epipolar l'. La línea epipolar es la proyección en la segunda imagen del rayo que va del punto x al centro C de la primer cámara. Así entonces, existe un mapeo de un punto en una imagen a su línea epipolar correspondiente en la otra imagen.

Como podemos ver en la Figura 2.3, las lineas epipolares en los planos de la imagen de dos cámaras convergentes se intersectan en el epipolo y son oblicuas, esto se aprecia claramente en la Figura 2.5. Sin embargo en las cámaras las cuales sus planos de la imagen son paralelos los epipolos están en el infinito y por lo tanto las lineas epipolares son paralelas, como se muestra en la Figura 2.6.

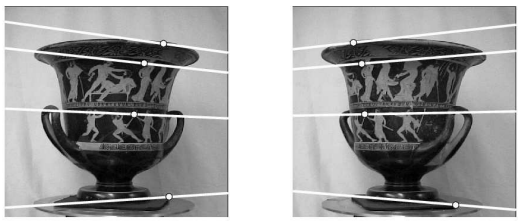


Figura 2.5: Lineas epipolares en cámaras convergentes.

2.1.3. Rectificación

Como se mencionaba en la *sección 2.1.1* al proceso de transformar las imágenes de las cámaras a una *posición estándar* se le conoce como *rectificación*. Este proceso nos permite obtener del sistema de visión estéreo imágenes que tienen la propiedad de que cada

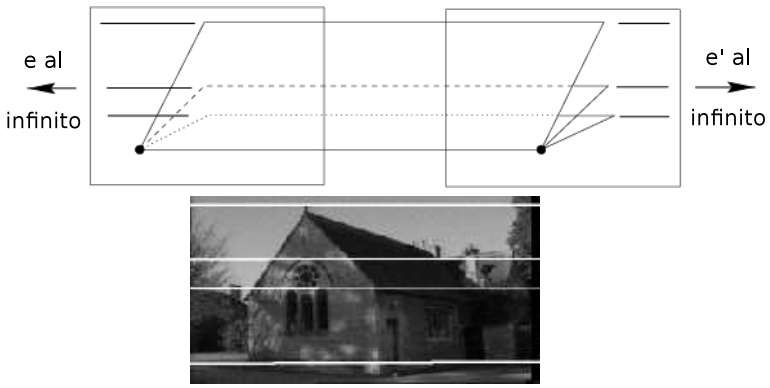


Figura 2.6: Lineas epipolares en cámaras paralelas.

punto en una imagen tiene su punto correspondiente en la misma linea horizontal en la otra imagen, es decir que las lineas epipolares que se describen en la *sección 2.1.2* se encuentren en la misma linea en las dos imágenes.

El proceso de calibración junto el proceso de rectificación determinan tanto los parámetros que las cámaras actualmente utilizan para producir las imágenes de entrada, como los de las imágenes en estado estándar utilizadas por el procesamiento estéreo, entre los cuales se encuentra la matriz de rectificación. La información de cómo son usados esos parámetros para transformar la imagen de entrada en la imagen rectificada se discuten a continuación.

En el presente trabajo de tesis se ha utilizado el modelo de Tsai [19] el cual incluye los parámetros intrínsecos que se muestran en el cuadro 2.1, estos parámetros describen las distorsiones introducidas en cada cámara individual por imperfecciones de los lentes y la colocación de los lentes. Como podemos ver se incluyen los parámetros de la distorsión radial y tangencial de los lentes, sin embargo sólo se utilizarán los dos primeros de la distorsión radial, puesto que los demás solo se utilizan en caso de lentes con distorsión muy alta. Los efectos más grandes vienen de la distorsión radial, en la cual la imagen se comprime hacia los bordes; y el descentrado de los lentes, en cual el centro de enfoque de los lentes no está en el centro de la matriz de la imagen.

Dado un punto en 3D $\mathbf{X} = (X, Y, Z)^T$ en las coordenadas de la cámara izquierda, se mapea a las coordenadas de la imagen usando los siguientes pasos. Primero, proyectamos

(x_0, y_0)	Centro de cámara
α_x, α_y	Distancia focal en terminos de dimensiones de los píxeles
$\kappa_1, \kappa_2, \kappa_3$	Parámetros de la distorsión radial
τ_1, τ_2	Parámetros de la distorsión tangencial

Cuadro 2.1: Parámetros intrínsecos de la cámara.

$\mathbf{X}$ al plano de la imagen normalizado usando las ecuaciones de perspectiva estándar

$$X_u = \frac{X}{Z} \qquad Y_u = \frac{Y}{Z}, \tag{2.8}$$

donde (X_u, Y_u) son las coordenadas de la imagen normalizadas y sin distorsión. Después, la distorsión radial de los lentes es modelada por

$$X_d = X_u(1 + \kappa_1 r^2 + \kappa_2 r^4), \qquad Y_d = Y_u(1 + \kappa_1 r^2 + \kappa_2 r^4) \tag{2.9}$$

donde (X_d, Y_d) son las coordenadas de la imagen con distorsión y $r^2 = X_d^2 + Y_d^2$. Finalmente, las coordenadas de la imagen con distorsión son mapeadas a las coordenadas (píxeles) de la pantalla (X_f, Y_f) usando

$$X_f = \alpha_x X_d + x_0 \qquad Y_f = \alpha_y Y_d + y_0, \tag{2.10}$$

donde α_x y α_y las distancias focales en terminos de la dimension de los píxeles en la dirección x y y respectivamente.

Esta última ecuación puede ser escrita en una forma estándar usando la matriz de cámara K, que se menciona en la ecuación 2.3, de la siguiente manera:

$$X_f = KX_d \tag{2.11}$$

Las ecuaciones anteriores son usadas para eliminar la distorsión de una imagen, a partir de una imagen de entrada distorsionada. Los detalles de cómo obtener los parámetros de distorsión se encuentran en el artículo de Tsai [19].

Ahora para que el procedimiento de búsqueda de correspondencia (*sección 2.1.5*) funcione adecuadamente, los planos de la imagen de cámara deben ser coplanares, y la geometría epipolar ajustada para que las líneas de barrido coincidan. Como se muestra en el cuadro 2.2, hay una transformación rígida entre las dos cámaras, con tres grados traslacionales y tres grados rotacionales de libertad. Es difícil de conseguir esta alineación

de manera mecánica a menos de tener un sistema de ajuste preciso incorporado a la cabeza estéreo. Afortunadamente, si las cámaras están cerca de su alineación ideal, es posible un ajuste digital de los parámetros. El procedimiento de calibración estima los parámetros externos y entonces estos pueden ser usados para rectificar las imágenes estéreo.

(T_x, T_y, T_z)	Localización del centro de proyección derecho en las coordenadas de la cámara izquierda
(R_x, R_y, R_z)	Rotación de las coordenadas del cuadro de la cámara derecha con respecto de la cámara izquierda

Cuadro 2.2: Parámetros extrínsecos del sistema estéreo.

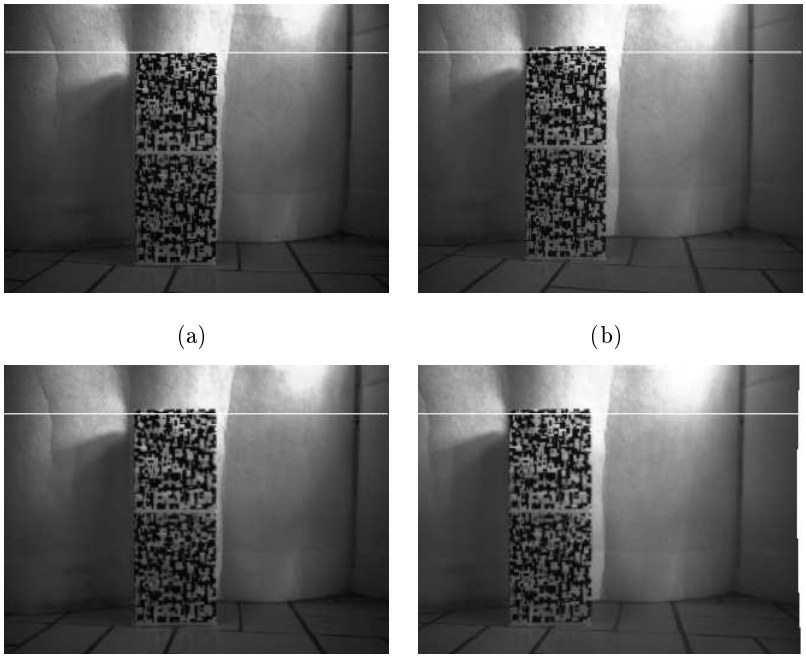
La rectificación estéreo compensa el hecho de que los ejes ópticos de las cámaras no sean paralelos. La salida del procedimiento de rectificación es una matriz que puede ser usada de manera efectiva para rotar los ejes ópticos de las cámaras para crear una configuración ideal estéreo. Se combina esta rectificación con la transformación descrita que corrige la distorsión.

La salida de la rectificación son dos matrices de proyección P_0 y P_1 de las cámaras izquierda y derecha rectificadas. El sistema de coordenadas 3D usado por las matrices de proyección es el sistema de coordenadas local izquierdo. En suma, la rectificación produce dos homografías H_0 y H_1 para transformar las imágenes izquierda y derecha de tal manera que sus ejes ópticos sean paralelos y los planos de la imagen sean coplanares.

En la Figura 2.7 vemos un ejemplo del efecto de la rectificación; las imágenes (a) y (b) son las imágenes originales de entrada sin rectificación, podemos ver claramente que la línea resaltada no contiene correspondencias a pesar de ser la misma línea de barrido; las imágenes (c) y (d) son las correspondientes imágenes rectificadas, podemos ver la corrección de la distorsión radial en la imagen (d) en la parte derecha, también es posible observar que los puntos correspondientes se encuentran en la misma línea de barrido en las dos imágenes.

2.1.4. Principios generales de la visión estéreo

La Figura 2.8 ilustra la relación de dos cámaras estéreo *ideales* (mediante el proceso de rectificación). El sistema coordenado global está centrado en el punto focal (centro de cámara) de la cámara izquierda. Se trata de un sistema de mano derecha, con Z positivo enfrente de la cámara, y X positivo hacia la derecha. El rayo principal de la cámara atraviesa



(a) (b)

(c) (d)

Figura 2.7: En (a) y (b) podemos ver las imágenes derecha e izquierda de entrada del sistema estéreo sin rectificación, en (c) y (d) se muestran las correspondientes imágenes rectificadas, en todas las imágenes se ha resaltado una línea para clarificar el efecto de la rectificación.

el plano de la imagen en las coordenadas x_0, y_0 en las dos cámaras. La distancia focal f es también la misma. La distancia entre los puntos focales, también conocida como *baseline* b, está alineada con el eje X.

El punto $\mathbf{X}$ es algún punto en la escena. Los puntos C_l y C_r son los centros de proyección, los puntos p_l y p_r son los puntos principales para la cámara izquierda y derecha, de igual forma I_l e I_r son los planos de la imagen izquierdo y derecho respectivamente. Dada la posición del punto x_l sobre el plano de la imagen izquierdo, $\mathbf{X}$ debe caer sobre algún punto en el rayo que se proyecta de C_l a x_l. Conociendo la posición de x_r sobre el plano de la imagen derecho, la posición de $\mathbf{X}$ puede ser determinada por triangulación al determinar el punto de intersección de los dos rayos.

En la Figura2.9 tenemos el esquema simplificado de la geometría estéreo. Formalmente, sea b la distancia entre C_l y C_r, Los puntos x_l y x_r la localización de $\mathbf{X}$ en los planos de la imagen de la cámara izquierda y derecha, sea f la distancia focal, y Z la distancia desde la linea base a $\mathbf{X}$, por la semejanza de los triángulos $(x_l, \mathbf{X}, x_r)$ y $(C_l, \mathbf{X}, C_r)$, tenemos

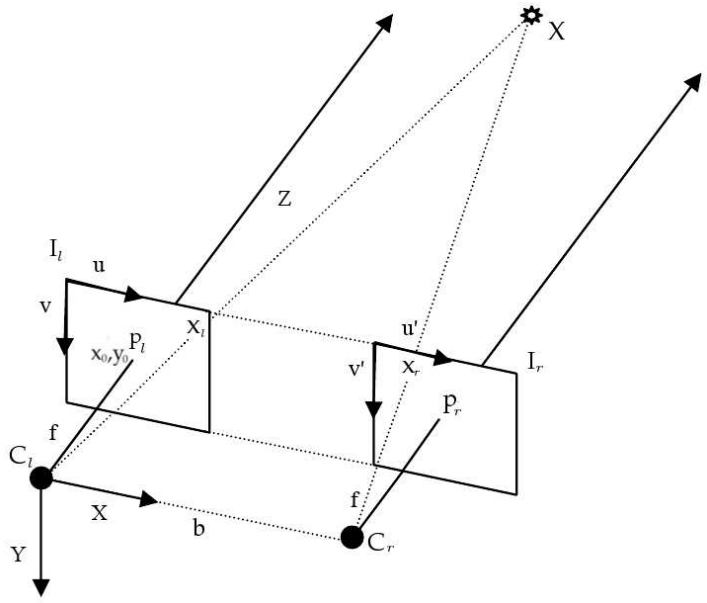


Figura 2.8: Geometría estéreo básica.

$$\frac{b-(x_l-x_r)}{Z-f}=\frac{b}{Z} \tag{2.12}$$

Si despejamos obtenemos la ecuación para Z

$$Z=\frac{fb}{d} \tag{2.13}$$

donde $d=x_l-x_r$.

Como d es la disparidad, es todo lo que necesitamos computar para determinar la posición de las características en la escena.

2.1.5. Cómputo de la disparidad

Las disparidades de todos los puntos de una imagen conforman el denominado *mapa de disparidad*, que puede ser representado como una imagen. Si el par de imágenes estéreo ha sido previamente *rectificado* entonces las líneas epipolares conjugadas se vuelven paralelas a la línea base del sistema estéreo. Esto significa que los puntos que se corresponden en ambas imágenes tienen la misma coordenada en el eje vertical y y reducen el problema de

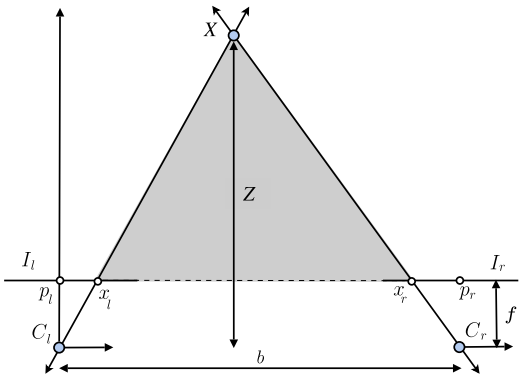


Figura 2.9: Principio de la visión estéreo.

correspondencia, mejor conocido como emparejamiento, a la búsqueda en una sola dimensión (eje horizontal x).

La principal guía para resolver el emparejamiento de un par de imágenes estéreo, se basa en la semejanza o la no semejanza. Características o áreas correspondientes deben ser similares (semejantes) en ambas imágenes del par estéreo.

Los métodos de emparejamiento pueden ser caracterizados como locales o globales. Los métodos locales tratan de emparejar pequeñas regiones de una imagen con la otra basados en las características intrínsecas de la región. Los métodos globales complementan los locales al considerar restricciones físicas como la continuidad de la superficie. Los métodos locales pueden ser clasificados entre sí emparejan *características* discretas entre las imágenes, o correlacionan una pequeña *área* [20].

En las técnicas basadas en la correlación de área, los elementos a comparar son ventanas de la imagen de dimensión fija, y el criterio de semejanza o criterio de correlación es una medida de la correspondencia entre las ventanas de las dos imágenes. El elemento correspondiente queda determinado por la ventana que maximiza el criterio de semejanza dentro de la región de búsqueda. El tamaño del área es un compromiso, puesto que las áreas pequeñas es más probable que sean similares en imágenes con diferentes puntos de vista, pero áreas grandes incrementan la relación señal/ruido. En contraste con los métodos basados en características, los basados en correlación de áreas producen resultados con mayor densidad.

El método de correlación de área tiene 5 pasos:

1. Corrección de la Geometría. En este paso, las distorsiones en las imágenes de entrada

son corregidas al transformarlas a una "forma estándar" (*rectificación*).

2. Transformación de la imagen. Un operador local transforma cada píxel en la imagen de escala de grises a una forma más apropiada, por ejemplo normalizándolo en base a la intensidad promedio local.

3. Correlación del área. Este es el paso de correlación en el cual cada pequeña área se compara con las otras áreas en su ventana de búsqueda.

4. Extracción del extremo. Es determinado el valor extremo de la correlación de cada píxel, lo cual origina la *imagen de disparidad*: cada valor de los píxeles es la disparidad entre las ventanas que mejor se emparejan entre la imagen izquierda y derecha.

5. Post-filtrado. Se aplican filtros que eliminan el ruido de la imagen de disparidad resultante.

La correlación de las áreas de la imagen son perturbadas por la iluminación, la perspectiva y las diferencias de la imagen. Los métodos de correlación usualmente los tratan de compensar al correlacionar no las intensidades en bruto de la imagen, sino alguna transformación de las intensidades. Entre los tipos básicos de transformación los mejores resultados parecen venir del criterio de diferencias absolutas (SAD) del laplaciano de la gaussiana (LOG) [20]. El laplaciano mide las intensidades de los bordes sobre alguna área suavizada por un filtro gaussiano. Típicamente la desviación estándar de la gaussiana es 1-2 píxeles.

El procedimiento de correlación opera como se describe a continuación. Dada una imagen I_i de dimensiones $N \times M$ y una ventana V_i de dimensiones $n \times m$ en torno a un píxel $I_i(x, y)$ de la imagen izquierda, se busca encontrar una ventana V_d sobre la imagen derecha I_d que sea *similar* a la anterior. Al encontrar la ventana de mayor similitud se dice que el píxel central $I_i(x, y)$ se corresponde con el píxel central $I_d(x', y)$ de la ventana derecha. Como las imágenes están rectificadas y se utiliza la restricción epipolar, la búsqueda se realiza sólo sobre la misma línea en la I_d desde la posición (x, y) hasta la posición $(x - d_{max}, y)$, donde d_{max}, representa la disparidad máxima. Se define $D(x, y, d)$ como el valor de la correspondecia más semejante para el píxel (x, y). Si se calcula $D(x, y, d)$ para cada píxel se obtiene una imagen denominada mapa de disparidad. Este mapa de disparidad tiene dimensiones $(N - n + 1) \times (M - m + 1 - d_{max})$.

Para decidir cual es la correspondencia más semejante se utiliza un criterio de semejanza ϕ, existen diversos tipos difiriendo entre sí en el costo computacional, la eficiencia, etc... [21]. El utilizado en este documento es el que a continuación se describe.

El criterio **suma de diferencias absolutas** SAD (sum of absolute differences) o L1 norm es un método fácil de implementar y que implica menor costo computacional, aspectos importantes si se considera la implementación en tiempo real. La formula que describe el grado de semejanza ϕ_{SAD} es:

$$\phi_{SAD(x,y,d)} = \sum_{j=1}^{m}\sum_{i=1}^{n} abs[V_d(i,j) - V_i(i,j)] \tag{2.14}$$

donde $V_i(i,j)$ es el píxel (i,j) en la ventana V_i de $n \times m$ con centro en el píxel (x,y) de la imagen I_i, de la misma manera se define el valor de $V_d(i,j)$, con la diferencia que el píxel central es $I_d(x+d,y)$.

El algoritmo 1 computa la disparidad. Para cada píxel en la imagen izquierda, el algoritmo trata de minimizar una función que definiremos como el grado de semejanza ϕ sobre la imagen izquierda sujeto a las restricciones de la geometría de la cámara.

Algoritmo 1 Cómputo de la disparidad [7]

1: Sea $D(x,y,d)$ la disparidad de cada píxel $I_l(x,y)$ a su correspondiente píxel en I_r
2: Sea $\phi_{(x,y,d)}$ la función que computa el grado de semejanza entre $I_l(x,y)$ y $I_r(x',y)$
3: **for** cada píxel x, y en I_l **do**
4: **for** cada emparejamiento posible x', y en I_r **do**
5: Computar $\phi_{(x,y,d)}$
6: **end for**
7: $D(x,y,d)$ sera la elección del x',y que tiene el mínimo valor de $\phi_{(x,y,d)}$
8: **end for**

La imagen de salida la cual tiene un amplio intervalo de valores usualmente contiene emparejamientos erróneos que deben ser filtrados. Entre diversas operaciones de filtrado se aplica el filtro de operador de interés [20] el cual da alta confianza a áreas que están texturizadas, ya que las áreas planas están comúnmente sujetas a emparejamientos ambiguos.

Finalmente, la imagen de disparidad puede ser procesada para proporcionar precisión de sub-píxeles, al tratar de localizar el pico de correlación entre píxeles. Esto incrementa

el intervalo de resoluciones disponibles sin demasiado trabajo adicional.

Los algoritmos estéreo típicamente buscan solo en una ventana de disparidades, por ejemplo 16 o 32 disparidades. En este caso, el conjunto de objetos que pueden ser determinados con éxito está restringido a cierto intervalo. El *horopter* es el volumen 3D que es cubierto por el rango de búsqueda del algoritmo estéreo. El horopter depende de los parámetros de la cámara y del baseline, del intervalo de búsqueda de disparidad y del X-offset. La Figura 2.10 muestra un horopter típico. EL algoritmo estéreo busca en un intervalo de 16-píxeles de disparidad para encontrar una correspondencia. Un objeto que tenga una correspondencia valida debe estar en la región entre los dos planos mostrados en la figura. El plano cercano tiene la más alta disparidad (15) y el más lejano la más baja (0).

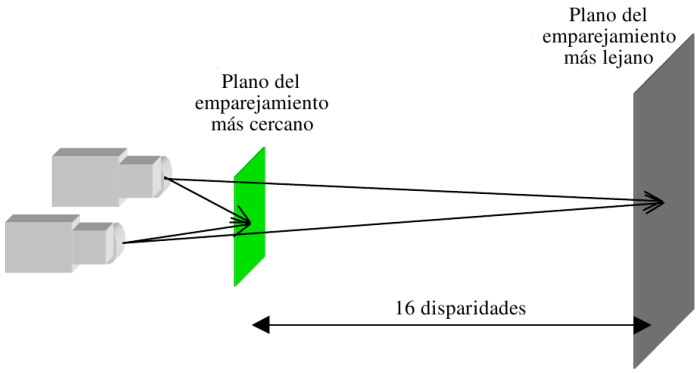


Figura 2.10: Planos del horopter para una búsqueda de 16-píxeles de disparidad.

La localización del horopter puede ser variada al cambiar el X-offset entre las dos imágenes, el cual esencialmente cambia la ventana de búsqueda para la correspondencia estéreo. El X-offset que pone el plano lejano del horopter en el infinito se denomina X_0. Con este desplazamiento, una disparidad de 0 indica un objeto infinitamente lejos.

El horopter debe ser ajustado de acuerdo a los requerimientos de la aplicación y este puede hacerse más grande por alguna combinación de lo siguiente:

- Decrementando el baseline.
- Decrementando la distancia focal (lentes de ángulo más amplio).
- Incrementando el ancho del píxel.
- Incrementando el tamaño de la ventana de búsqueda de la disparidad.

Las tres primeras opciones cambian la geometría de la cámara, y por tanto tienen su efecto correspondiente en la intervalo de resolución, el cual se decrementa. La manera efectiva de incrementar el horopter y mantener la resolución es incrementar el tamaño de la ventana de búsqueda de la disparidad.

2.2. Modelo Directo

En los años recientes el concepto de modelo interno, un sistema que imita el comportamiento de un proceso natural, ha emergido como un concepto teórico importante en el control motriz. Un tipo de modelo interno que es la representación causal del aparato motriz son los modelos directos. Este modelo tiene como objetivo imitar o representar el comportamiento normal del sistema motriz en respuesta a los comandos motrices de salida.

Un modelo directo incorpora conocimiento acerca de cambios sensoriales producidos por acciones autogeneradas de un agente. El termino modelo directo proviene del estudio de cómo controlar el comportamiento de los sistemas dinámicos. El principio está ilustrado en la Figura 2.11 [22]. Las flechas continuas indican el ciclo en el cual se traduce el comando motriz en la salida motriz, ésta tiene algún efecto sobre el mundo, causando algunos estímulos sensoriales, que el sistema puede procesar para generar el siguiente comando. El modelo directo es un ciclo interno que toma el comando motriz y la salida del procesamiento sensorial y predice la entrada sensorial esperada, la cual puede ser usada para modular el procesamiento sensorial de la entrada actual.

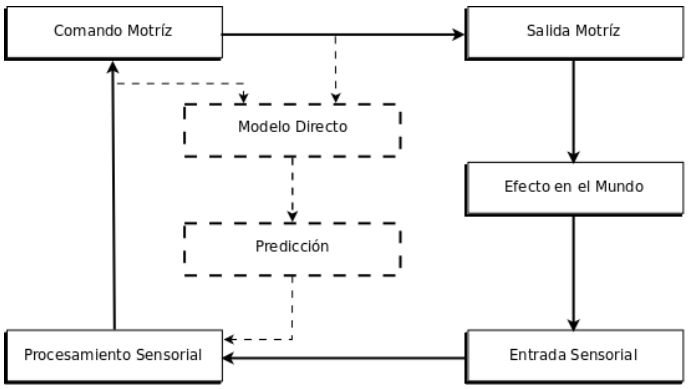


Figura 2.11: Representación esquemática del sistema sensori-motor con un modelo directo.

Un ejemplo clásico es considerar que a pesar de que nuestros ojos siempre registran

movimientos, y eso hace que la imagen en la retina se mueva, nosotros percibimos un mundo estable, esto se debe en parte a que nuestro cerebro compensa el movimiento de la imagen, el cual es predecible a partir del movimiento del ojo conocido por el sistema mismo. Otro ejemplo, un modelo directo de la dinámica del brazo, el cual podría tener como entrada el estado actual (por ejemplo los ángulos de las uniones y las velocidades) y los comandos motrices expedidos por un controlador, y producir como salida una estimación del nuevo estado. Este modelo por lo tanto captura el comportamiento de la transición de estados del brazo en respuesta a la salida motriz. Por estado nos referimos a la posición actual y velocidad del aparato motriz, o en términos más generales, a un conjunto de parámetros los cuales tomados junto con el conocimiento de las entradas y la dinámica del sistema determinan su comportamiento futuro.

Como los modelos directos son causales también son funciones bien definidas en las cuales los mapeos son uno a uno o varios a uno. Estos modelos han demostrado ser altamente convenientes en un amplio intervalo de situaciones de control motriz.

Los modelos directos pueden tener diversos usos, a continuación se enuncian algunos para ilustrar la utilidad de estos modelos.

Un modelo directo es un ingrediente clave en un sistema que usa salida motriz para anticipar y cancelar los efectos sensoriales del movimiento. Las señales sensoriales surgen en la periferia por dos motivos: aquellas que son el resultado de las influencias del entorno en el cuerpo, y aquellas que son autogeneradas por los movimientos. Las primeras se denominan aferencias, mientras el segundo tipo de señales sensoriales son conocidas como *re*aferencias ya que son las consecuencias sensoriales del movimiento. Aunque las aferencias y las reaferencias tienen distintas causas ambas son percibidas por los mismos canales sensoriales. Desde un punto de vista comportacional puede ser necesario distinguir las dos causas entre las señales especialmente para monitorear los cambios en el mundo externo separadamente de aquellos resultado del movimiento propio.

Como un ejemplo consideremos el problema de mover la mano sobre un objeto sobre una mesa y estimar sin la ayuda de la visión si el objeto se esta moviendo. La velocidad de deslizamiento (la velocidad del objeto a través de la palma de la mano) es la suma de la velocidad de la mano y la velocidad del objeto en el mundo exterior. Entonces para decidir si el objeto se encuentra en movimiento, es necesario primeramente remover la componente de deslizamiento generada por el movimiento del brazo. Sin embargo, una copia de la eferencia, es decir una copia del comando motriz descendente que actúa sobre el sistema sensorimotriz,

no puede por si misma proveer esta información, pues está es una señal motriz predictiva de la activación muscular, mas que una entrada sensorial. Al generar un estimado de las consecuencias sensoriales de un comando motriz, un modelo directo interno puede ser usado para cancelar las señales sensoriales reaferentes, y entonces permitir recuperar las señales externas relativas al entorno.

Uno de los problemas fundamentales a los cuales se enfrenta el sistema nervioso central en el contexto del control es que el objetivo y los resultados de un movimiento son a menudo definidos en coordenadas relativas a la tarea. Por ejemplo, cuando se trata de alcanzar un objetivo visual el objetivo es inicialmente especificado en un marco visual. Durante el movimiento, cualquier error en el comando motriz causa un error visual. Un problema básico existe, por lo tanto, en traducir esos objetivos relativos a la tarea y errores a las señales intrínsecas apropiadas (comandos motrices y errores motrices) las cuales son requeridas para actualizar el controlador. La relación directa entre las señales motrices y las señales sensoriales puede ser capturada por un modelo directo. Se ha demostrado que un modelo directo puede ser usado para transformar errores entre la salida sensorial actual y la deseada de un movimiento en los correspondientes errores en el comando motriz, y de este modo proveer las señales apropiadas para el aprendizaje motriz [23].

De igual manera un modelo directo puede ser usado para estimar el estado del sistema algún tiempo en el futuro. Tal predicción puede ser usada en al menos dos contextos.

La primera manera es el modelo predictivo de control. Se puede tomar la idea básica de un circuito de retroalimentación interna y extrapolarla en el comportamiento futuro del sistema. Si el valor de referencia que se esta tratando de alcanzar con el controlador se conoce de antemano, entonces el estado estimado del desempeño futuro del sistema puede ser comparado de antemano, y los errores predichos corregidos antes de que ocurran.

La segunda forma es el ensayo mental y la planeación. El ensayo mental puede ser pensada como el realizar los movimientos mentalmente sin moverse. Es conocido que el ensayo mental puede conducir a un desempeño mejorado, y además permite monitorear el desempeño y que el aprendizaje motriz tome lugar en la ausencia de una acción real. Durante tal practica mental puede usarse un modelo directo para predecir la salida de una o una serie de acciones: un estimado de un estado futuro (el cual por supuesto pudo ser generado por un modelo directo) y los comandos motrices apropiados podrían ser dirigidos al modelo y bloqueados antes de llegar al aparato motor. Por lo tanto un uso iterativo o recursivo del modelo directo podría permitir el ensayo mental y las imágenes mentales de

distantes estados motrices arbitrarios. Basado en la relación entre el movimiento deseado y su salida predicha dada por el modelo, un controlador podría seleccionar entre las posibles acciones, o podría el mismo adaptarse. Por lo tanto, un modelo directo podría estar involucrado en planeación motriz. Por ejemplo, con el fin de tomar una taza, el planificador debe desarrollar un programa motriz adecuado que involucre el movimiento de la mano y del brazo. Sin embargo, si la taza se encuentra demasiado lejos del cuerpo, el plan puede incluir el movimiento del tronco y posiblemente incluso la locomoción. Al probar internamente el plan desarrollado, o los planes alternativos, vía un modelo directo, seria posible evaluar su utilidad. Por consiguiente, el planificador podría inicialmente probar el plan "extender brazo", pero cuando el modelo directo prediga que la mano caería lejos aun de la posición de la taza, el planificador podría al menos rechazar tal plan, a pesar de que el modelo directo no puede por sí solo proveer un plan mejor.

Consideremos un ejemplo simple, tal como el control de navegación de un coche, donde el objetivo es controlar la velocidad del vehículo con un comando motriz (normalmente dado por el acelerador o el freno) que cambia la aceleración. El cambio en la velocidad de la rueda (la salida del motor) será una función compleja de factores como las características del motor, la velocidad actual, y el acoplamiento del motor a las ruedas. El cambio en la velocidad del coche (el efecto de la salida del motor en el mundo) también dependerá de la superficie de la carretera, la resistencia al viento del vehículo, etc. La velocidad real puede ser medida por un sensor como un tacómetro. En teoría, todos estos factores podrían estar representados en el modelo directo que toma el comando motriz y la entrada sensorial como entrada y predice la velocidad, mediante la simulación de los procesos correspondientes, que se medirá por el tacómetro (la entrada sensorial al tiempo siguiente).

¿Porque no solo dejar funcionar al sistema y observar las consecuencias? Hay varias ventajas de hacer la predicción. Una es el potencial tiempo de retraso involucrado en la retroalimentación sensorial. Por ejemplo, si el controlador espera hasta que el tacómetro alcance el límite de velocidad antes de dejar de acelerar, el ajuste podría llegar demasiado tarde para evitar que se sobrepase el límite de velocidad. Otra es que la comparación entre la predicción y la observación se puede utilizar para distinguir las perturbaciones externas de la retroalimentación esperada. Por ejemplo, si el coche tiene un neumático desinflado, no queremos que el controlador trate de continuar incrementando la salida tratando de alcanzar la velocidad deseada a pesar de la fricción adicional, sino más bien que detecte que algunos cambios significativos en la dinámica del sistema se han producido. Las diferencias entre

la predicción y la retroalimentación también se pueden utilizar para aprender y mejorar la generación de las señales de control. Otra función de la predicción es que se puede ejecutar fuera de línea para probar si una entrada de control particular, es viable o lleva a malas consecuencias.

Una de las características de un modelo directo es la adaptabilidad. Un modelo directo el cual captura la dinámica del brazo de un niño de tres años es poco probable que sea usado en un adulto. Por ello un modelo directo aumenta su utilidad al ser adaptable. En general las señales requeridas para entrenar y actualizar el modelo interno pueden ser fácilmente generadas. Cualquier discrepancia entre la salida del modelo y la salida actual del sistema puede ser usada para actualizar el modelo usando alguna de las técnicas de aprendizaje supervisado. Esta estrategia ha demostrado ser eficaz utilizando redes neuronales artificiales para la producción de modelos directos razonablemente exactos.

Las evidencias que apoyan a los modelos directos son varias aunque de manera indirecta, sin embargo existen algunos experimentos que apuntan a la existencia y uso de los modelos directos internos [23]. Por mencionar alguno podemos referirnos al siguiente: cuando movemos nuestro brazo en la ausencia de retroalimentación visual, hay métodos que el sistema nervioso central puede usar para obtener un estimado del estado actual, la posición y la velocidad, de la mano. El sistema podría hacer uso de la entrada sensorial propioceptiva, podría hacer uso de la salida motriz integrada, o podría combinar estos dos canales de información por medio del uso de un modelo directo. Wolpert [24] estudio la tarea de integración sensorimotriz en la cual varios sujetos estimaban la localización de su mano al final de varios movimientos hechos en la oscuridad. Los sujetos típicamente tenían pequeños errores al reportar la posición de la mano, los cuales variaban con la duración del movimiento. Creó un modelo directo simulado que tenia como entrada el comando motriz y el estado actual estimado y utilizaba un filtro de Kalman, tras lo cual comparó las curvas de estimación de posición de los sujetos con las de la simulación, y en ellas se nota una clara similitud.

2.3. Redes Neuronales

Las redes de neuronales artificiales (denominadas habitualmente como RNA) son un modelo computacional de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso de los animales. Las redes neuronales consisten en una

simulación de las propiedades observadas en los sistemas neuronales biológicos a través de modelos matemáticos recreados mediante mecanismos artificiales.

Una red neuronal se compone de unidades abstractas llamadas neuronas. En la figura 2.12 se muestra una neurona con n entradas, cada canal de entrada i puede transmitir un valor real x_i. Cada neurona recibe las entradas a través de interconexiones y emite una salida. Esta salida viene dada por lo siguiente [25]:

1. Un conjunto de pesos sinápticos $W_j = (w_{1j}, w_{2j}, ..., w_{nj})$ para cada neurona j; donde w_{ij} es el peso asociado a una entrada i de la j-ésima neurona. Cada una de las entradas es multiplicada por su correspondiente peso sináptico. Estos valores son los parámetros libres que se irán ajustando durante el entrenamiento.

2. Una función de entrada u_i (también conocida como función de excitación), $u_i = \sum_{j=0}^{n} w_{ji} x_j$ si el peso w_i es positivo, la conexión se denomina excitatoria; si es negativo, se denomina inhibitoria.

3. Una función de activación f, que se aplica sobre la entrada. Generalmente viene dada por la interpretación que queramos darle a dichas salidas. Algunas de las más utilizadas son la función sigmoidea (para obtener valores en el intervalo $[0, 1]$) y la tangente hiperbólica (para obtener valores en el intervalo $[-1, 1]$).

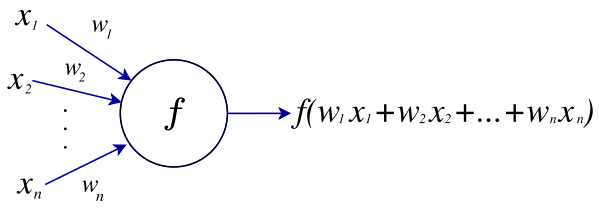


Figura 2.12: Una neurona abstracta.

Si concebimos a cada neurona en una red neuronal como una función primitiva capaz de transformar su entrada en una salida definida con precisión, entonces una RNA no es otra cosa que una red de funciones primitivas. Los diferentes modelos de RNA difieren principalmente en los supuestos acerca de las funciones primitivas usadas, el patrón de interconexión y en el momento en que se transmite la información [26].

La función implementada por una red neuronal sería llamada la función de la red. La selección de los valores de los pesos w_i producen diferentes funciones de red. Sin embargo, tres elementos son particularmente importantes en cualquier modelo de RNA:

- La estructura de los nodos
- La topología de la red
- El algoritmo de aprendizaje usado para encontrar los pesos de la red y los parámetros que pueden definir su funcionamiento.

De acuerdo a la topología en una RNA se definen tres tipos básicos [27]:

- Dos tipos de redes de propagación hacia delante o acíclicas en las que todas las señales van desde la capa de entrada hacia la salida sin existir ciclos, ni conexiones entre neuronas de la misma capa.
  - Monocapa, como el perceptrón.
  - Multicapa, como el perceptrón multicapa.
- Las redes recurrentes que presentan al menos un ciclo cerrado de activación neuronal. Por ejemplo las redes Elman [28] y de Hopfield [29].

De acuerdo al tipo de aprendizaje las redes se clasifican en:

- Aprendizaje supervisado: necesitan un conjunto de datos de entrenamiento previamente clasificado o cuya respuesta objetivo se conoce. Por ejemplo: el perceptrón simple, la red Adaline, el perceptrón multicapa, etc.
- Aprendizaje no supervisado o autoorganizado: el sistema se supone que estadísticamente debe descubrir los aspectos más destacados de la población de entrada. A diferencia del aprendizaje supervisado, no existe un conjunto a priori de categorías en las que se clasifican los patrones, más bien el sistema debe desarrollar su propia representación de los estímulos de entrada. Por ejemplo los mapas auto-organizados de Kohonen [30].

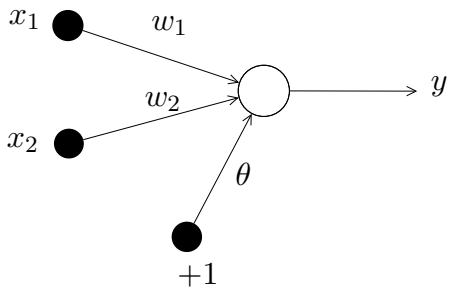


Figura 2.13: Perceptrón con dos entradas y una salida.

2.3.1. Perceptrón

El perceptrón es una RNA de una sola capa con propagación hacia adelante, la cual computa funciones simples y que tiene el problema de no poder computar problemas que no sean linealmente separables.

En la Figura 2.13 podemos ver un ejemplo de un perceptrón con 2 entradas y una salida.

La entrada de la neurona es la suma ponderada de las entradas mas el termino de bias θ. La salida y de la red esta formada por la activación de la neurona de salida, la cual está en función de la entrada:

$$y = f(\sum_{i=1}^{n} w_i x_i + \theta) \tag{2.15}$$

Para ajustar los pesos y el bias al valor correcto se utiliza la regla de aprendizaje perceptrón. Este método es un procedimiento que ajusta los pesos. En términos generales lo que hace es presentar un patrón de entrenamiento a la red, para cada peso se computa el nuevo valor sumando un incremento al valor anterior para corregirlo, el bias es actualizado de la misma manera:

$$w_i(t+1) = w_i(t) + \Delta w_i(t) \tag{2.16}$$

$$\theta(t+1) = \theta(t) + \Delta\theta(t) \tag{2.17}$$

El problema de aprendizaje puede ser formulado en términos de encontrar $\Delta w_i(t)$ y $\Delta\theta(t)$.

Supongamos que tenemos un conjunto de ejemplos de aprendizaje consistentes de un vector de entrada χ, la salida deseada $d(\chi)$ y un factor de aprendizaje η. La regla de aprendizaje perceptrón [27] puede resumirse como sigue:

1. Inicialización de los pesos a un valor aleatorio.
2. Seleccionar un vector de entrada χ del conjunto de ejemplos de entrenamiento.
3. Si $y \neq d(\chi)$ (el perceptrón esta dando una respuesta incorrecta), modificar todas las conexiones w_i de acuerdo a $\Delta w_i = \eta(d(\chi) - y)x_i$, en caso contrario terminar.
4. Volver a 2.

Podemos ver un diagrama que ejemplifica el proceso de aprendizaje en este tipo de redes en la Figura 2.14.

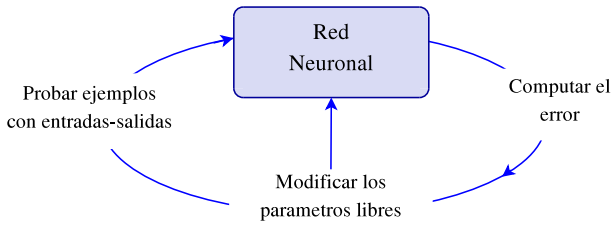


Figura 2.14: Proceso de aprendizaje supervisado en una RNA.

Consideremos ahora redes con más de una capa. El caso más común involucra una sola capa oculta. La ventaja de adicionar capas ocultas es que aumenta el espacio de hipótesis que la red puede representar.

Por ejemplo si pensamos en cada unidad oculta como un perceptrón que representa una función umbral sigmoidal en el espacio de entrada como se muestra en la Figura 2.15.a. Podemos realizar una combinación lineal de varias de tales funciones en la unidad de salida, digamos, adicionando dos funciones en oposición para obtener una función cresta como se muestra en la Figura 2.15.b.

2.3.2. Retropropagación del error

Al estudiar las redes multicapa se nos presenta la pregunta ¿cómo ajustar los pesos de la entrada a las capas ocultas? La idea central detrás de la solución es que los errores

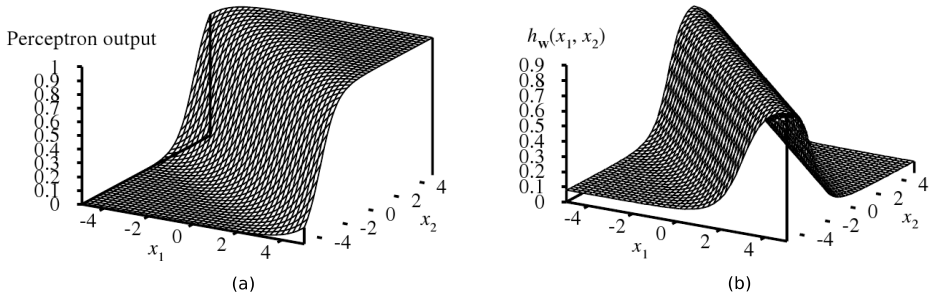


Figura 2.15: Resultantes de la combinación de 2 y 4 unidades perceptrón respectivamente.

de la capa oculta son determinados al propagar hacia atrás los errores de las unidades en la capa de salida. Por esta razón el método es regularmente llamado regla de aprendizaje de retropropagación del error (error backpropagation).

Para ayudar a clarificar la descripción que se presenta a continuación, podemos observar la Figura 2.16 en la cual se muestra una red perceptrón de 2 capas, con n entradas, capa oculta de m unidades y l salidas. Cada neurona ha sido separada en dos componentes: la función de entrada representada como una sumatoria y la función de activación que en este caso es una sigmoidal. Cabe aclarar que en la Figura 2.16 no se han mostrado todos los pesos $w_{ij}^{(k)}$ sino solo algunos de manera representativa, donde k es la capa e i y j son las neuronas conectadas por la arista con el peso mencionado.

Es necesario definir un conjunto de p patrones de entrenamiento $(\mathbf{x}, t(\mathbf{x}))$, donde $\mathbf{x}$ es un vector de valores de entrada y $t(\mathbf{x})$ es el vector que indica las salidas deseadas.

Una vez inicializados los pesos $w_{ij}^{(k)}$ a un valor aleatorio, la regla de aprendizaje de retropropagación del error dice:

1. Computo hacia adelante.
2. Propagar hacia atrás a la capa de salida.
3. Propagar hacia atrás a la capa oculta.
4. Actualizar los pesos.

El algoritmo se detiene cuando el valor del error $E = \frac{1}{2}\sum_{i=1}^{l}(o_i^{(2)} - t_i(\mathbf{x}))^2$ (Figura 2.17) es suficientemente pequeño, donde $o_i^{(2)}$ es la salida de la red.

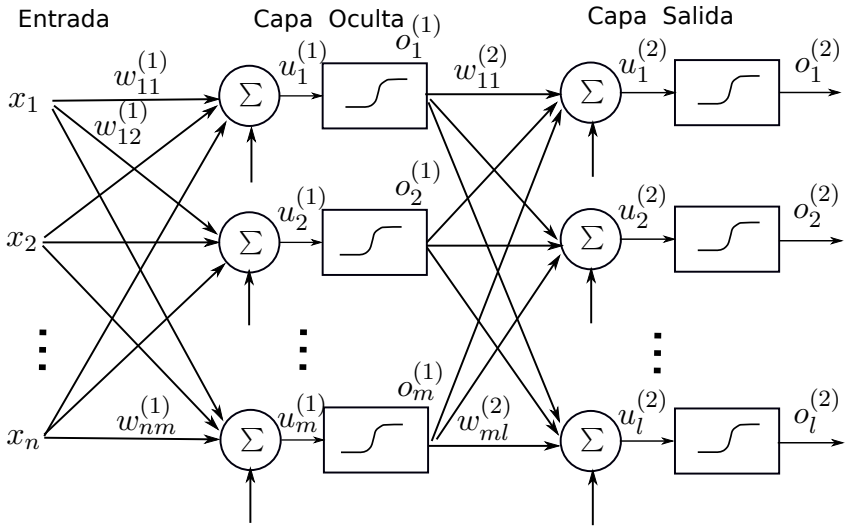


Figura 2.16: Perceptrón multicapa.

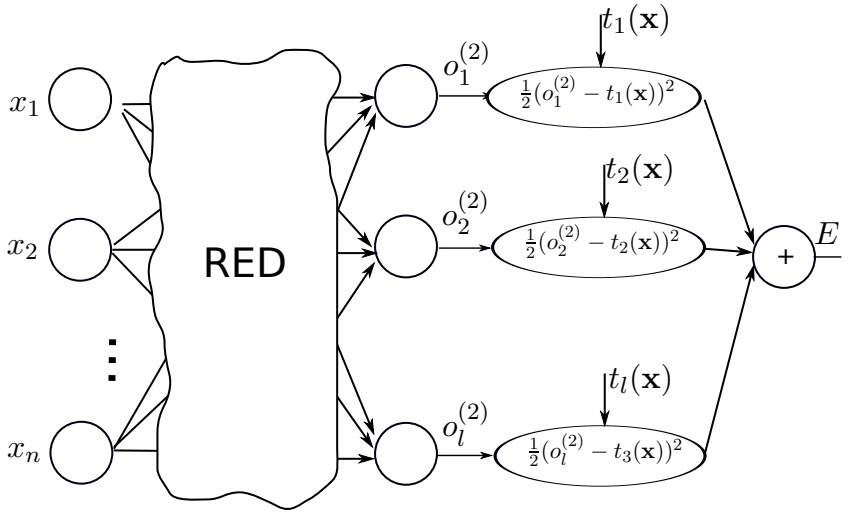


Figura 2.17: Computo de la función de error.

Primer paso: Computo hacia adelante. Se presenta un vector de entrada $\mathbf{x}$ a la red. Se computan los valores $o_i^{(1)}$ y $o_i^{(2)}$.

Segundo paso: Propagación hacia atrás a la capa de salida. Hay que buscar los valores de $\frac{\partial E}{\partial w_{ij}^{(2)}}$, los cuales por regla de la cadena deben ser:

$$\frac{\partial E}{\partial w_{ij}^{(2)}} = \frac{dE}{do_j^{(2)}} \cdot \frac{do_j^{(2)}}{du_j^{(2)}} \cdot \frac{du_j^{(2)}}{w_{ij}^{(2)}} \tag{2.18}$$

Lo cual nos da para la función sigmoide:

$$\frac{\partial E}{\partial w_{ij}^{(2)}} = (o_j^{(2)} - t_j(\mathbf{x})) \cdot o_j^{(2)}(1 - o_j^{(2)}) \cdot o_i^1$$

Llamamos al error propagado a la capa de salida $\delta_j^{(2)}$:

$$\delta_j^{(2)} = o_j^{(2)}(1 - o_j^{(2)})(o_j^{(2)} - t_j(\mathbf{x}))$$

Con esto nos queda:

$$\frac{\partial E}{\partial w_{ij}^{(2)}} = \delta_j^{(2)} o_i^1 \tag{2.19}$$

Tercer paso: Propagación hacia atrás a la capa oculta. Ahora necesitamos el valor $\frac{\partial E}{\partial w_{ij}^{(1)}}$. Cada unidad j de la capa oculta está conectada a cada unidad l de la capa de salida con una arista con peso $w_{jq}^{(2)}$, para $q = 1, ..., l$, y tenemos los valores del error propagado a la capa de salida $\delta_q^{(2)}$ con lo cual tenemos:

$$\frac{\partial E}{\partial w_{ij}^{(2)}} = \sum_{q=1}^{l} w_{jq}^{(2)} \delta_q^{(2)} \cdot \frac{do_j^{(1)}}{du_j^{(1)}} \cdot \frac{du_j^{(1)}}{w_{ij}^{(1)}} \tag{2.20}$$

Llamamos $\delta_j^{(1)}$ a:

$$\delta_j^{(1)} = o_j^{(1)}(1 - o_j^{(1)}) \sum_{q=1}^{l} w_{jq}^{(2)} \delta_q^{(2)}$$

Con esto nos queda:

$$\frac{\partial E}{\partial w_{ij}^{(1)}} = \delta_j^{(1)} x_i \tag{2.21}$$

Cuarto paso: Actualización de los pesos. Una vez calculados los valores de las derivadas parciales solo basta actualizar los pesos, primero calculamos las correcciones a los pesos:

$$\Delta w_{ij}^{(k)} = -\eta \frac{\partial E}{\partial w_{ij}^{(k)}} \tag{2.22}$$

donde η es el factor de aprendizaje.

Se denomina entrenamiento *on-line* si se actualizan los pesos después de la presentación de cada patrón de entrenamiento. Si por el contrario se actualizan los pesos después de cada *época* (una época consiste de presentar todos los p patrones de entrada) entonces se trata de un entrenamiento *off-line* y el valor $\Delta w_{ij}^{(k)}$ se calcula como:

$$\Delta w_{ij}^{(k)} = \Delta_1 w_{ij}^{(k)} + \Delta_2 w_{ij}^{(k)} + ... + \Delta_p w_{ij}^{(k)} \tag{2.23}$$

La actualización de los pesos se hace de la siguiente manera:

$$w_{ij}^{(k)} = w_{ij}^{(k)} + \Delta w_{ij}^{(k)} \tag{2.24}$$

Algoritmo RPROP

Como podemos ver la elección del factor de aprendizaje η, la cual escala la derivada, tiene un efecto importante en el tiempo necesario hasta que se alcanza la convergencia. Si este valor es demasiado pequeño, se necesitan demasiados pasos hasta alcanzar una solución aceptable; por el contrario si se elige un factor de aprendizaje grande este posiblemente llevara a una oscilación, evitando que el error decrezca hasta el valor deseado.

Varios algoritmos han sido propuestos para tratar con el problema de la actualización de los pesos de manera adecuada haciendo algún tipo de adaptación de parámetros durante el aprendizaje. Estos pueden separarse en 2 categorías: estrategias globales y locales. Las técnicas de adaptación global hacen uso del conocimiento del estado de la red completa para modificar parámetros globales, mientras las estrategias locales usan solamente información especifica de un peso para adaptar solo los parámetros de ese peso sináptico.

La mayoría de los algoritmos adaptativos globales y locales realizan una modificación del factor de aprendizaje de acuerdo al comportamiento observado de la función de error. La razón principal de que el algoritmo RPROP sea más eficiente en el tiempo está cimentada en el concepto de la 'adaptación directa' del tamaño de la actualización de los

pesos, en contraste con todos los otros algoritmos, que solo usan el signo de la derivada parcial para realizar tanto el aprendizaje como la adaptación.

RPROP debe ser entendido por *"resilient propagation"* y es un esquema de aprendizaje que realiza adaptación directa del incremento del peso basado en la información local del gradiente [31].

El algoritmo introduce para cada peso su valor de actualización individual Δ_{ij}, el cual solamente determina el tamaño de la actualización del peso. Este valor de actualización adaptativo evoluciona durante el proceso de aprendizaje basado en su visión local de la función de error E, de acuerdo a la siguiente regla de aprendizaje:

$$\Delta_{ij}^{(t)} = \begin{cases} \eta^+ * \Delta_{ij}^{(t-1)} & \text{, if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ \eta^- * \Delta_{ij}^{(t-1)} & \text{, if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ \Delta_{ij}^{(t-1)} & \text{, else} \end{cases} \tag{2.25}$$

donde $0 < \eta^- < 1 < \eta^+$.

La regla de adaptación funciona como sigue: cada vez que la derivada parcial del peso w_{ij} correspondiente cambia su signo, indica que la última actualización fue demasiado grande y el algoritmo ha saltado sobre un mínimo local y el valor de actualización Δ_{ij} es decrementado por el factor η^-. Si la derivada retiene su signo, el valor de actualización es ligeramente incrementado en función de acelerar la convergencia en superficies llanas.

Una vez que el valor de actualización para cada peso es adaptado, la actualización del peso sigue una regla simple: si la derivada es positiva (aumento del error), el peso es disminuido por su valor de actualización, si la derivada es negativa, el valor de actualización se suma:

$$\Delta w_{ij}^{(t)} = \begin{cases} -\Delta_{ij}^{(t)} & \text{, if } \frac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ +\Delta_{ij}^{(t)} & \text{, if } \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ 0 & \text{, else} \end{cases} \tag{2.26}$$

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)} \tag{2.27}$$

Hay una excepción: si la derivada parcial cambia de signo, la actualización del peso se revierte:

$$\Delta w_{ij}^{(t)} = -\Delta w_{ij}^{(t-1)}, \text{ if } \frac{\partial E}{\partial w_{ij}}^{(t-1)} * \frac{\partial E}{\partial w_{ij}}^{(t)} < 0 \tag{2.28}$$

Los valores de actualización y los pesos se cambian cada vez que el conjunto patrones completo ha sido presentado a la red (aprendizaje *off-line*).

Una de las principales ventajas del RPROP es el hecho de que para varios problemas no es necesario hacer una elección de parámetros para obtener una convergencia cercana a la óptima [31]. Los parámetros recomendados son:

- Los valores de actualización son inicializados a un valor $\Delta_0 = 0.1$.
- El rango de los valores de actualización esta restringido a un limite superior de $\Delta_{max} = 1.0$ y un limite inferior de $\Delta_{min} = 1e^{-6}$.
- El factor de incremento se fija a $\eta^+ = 1.2$ y el factor de decremento a $\eta^- = 0.5$.

Capítulo 3

Metodología de solución

Como se menciono anteriormente, el objetivo principal del trabajo es el de dotar a un agente con una noción de distancia a los objetos que lo rodean. Esta noción deberá estar relacionada a la arquitectura propia del robot, a sus características físicas y capacidades motrices.

La metodología para solucionar el problema es la siguiente:

- Utilizando una cámara estéreo generar el mapa de disparidad. Para generar dicho mapa se utilizará la herramienta de calibración, rectificación y calculo de disparidad del software SVS desarrollado por Kurt Konolige [20].
- Aparte del sistema de visión estéreo se utilizan sensores táctiles, a fin de poder detectar colisiones con objetos, dichos sensores son emulados por un sistema de sonares umbralizados.
- Para el problema de la extracción de la información del mapa de disparidad se utiliza un modelo directo, que formará una representación multimodal entre la percepción visual, táctil y los comandos motrices, y que permite hacer predicciones de posibles colisiones a corto plazo, que encadenadas son predicciones a mediano plazo, éste es el indicativo para corroborar que se está efectuando una asociación de la información del mapa de disparidad con las propiedades del robot.
- El modelo directo se obtiene al entrenar una red neuronal artificial con datos que provienen del robot real situado en una arena, la red es entonces probada con trayectorias que no se han utilizado durante el entrenamiento.

El sistema completo se implementa en el robot para resolver una tarea de evasión de obstáculos.

3.1. Descripción del hardware

Los componentes más importantes que integran la plataforma del sistema son el robot, el sistema de visión estéreo y el sistema de sonares (parachoques), a continuación se describen las características más importantes de cada uno de estos elementos.

El robot utilizado es el modelo Pioneer 3-DX (Figura 3.1) el cual es un robot orientado a la investigación académica. Al ser un robot versátil, confiable y durable se ha convertido en uno de los robots móviles más utilizados en la investigación. La base de la plataforma P3-DX esta ensamblada con motores con enconders de 500 marcas, dos ruedas diferenciales de 19cm, 8 sensores ultrasónicos (sonares) en la parte frontal, un microcontrolador de alto desempeño con un software de control embebido de 32-bit Renesas SH2-7144 RISC y una computadora interna con Linux Suse, el software Advanced Robotics Interface for Applications (ARIA) y ARNetworking, desarrollado bajo la licencia pública GNU con librerías para C++, Java y Python.

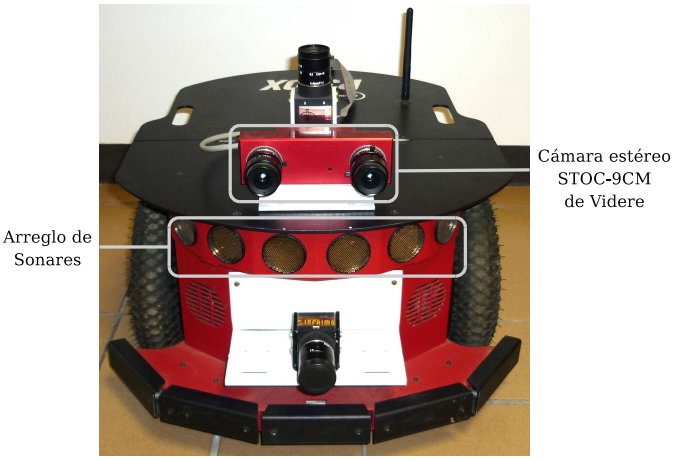


Figura 3.1: Robot Pioneer 3-DX con la cámara STOC-9CM de videre.

La base Pioneer 3 DX puede alcanzar velocidades de hasta 1.6 metros por segundo y cargar hasta 23 kg.

La adquisición de imágenes se realiza mediante la cámara digital estéreo STOC-

9CM de la empresa Videre Design (Figura 3.1). Esta cámara tiene función de captura en color o monocromática, cuenta con una resolución máxima de 640H x 480V pixeles, la linea base es de 9cm, con lente de 1.4/6.0 mm $57,3°$ HFOV (campo de visión horizontal) x 44° VFOV (campo de visión vertical), cuenta también con interfase al software SVS para análisis estéreo.

El microcontrolador del sistema de la cámara STOC puede realizar el procesamiento estéreo rápidamente y con bajo consumo de energía. La interfase estándar FireWire entrega vídeo y la información de disparidad sobre el mismo cable, para el cálculo de la disparidad utiliza ventanas de 15 píxeles de tamaño y la entrega en un formato de 10 bits.

El sistema de sonares (Figura 3.1), que harán de parachoques para nuestro sistema, está compuesto de 8 unidades ultrasónicas de la marca SensComp serie 600 que tienen como principales características una frecuencia de ultrasonido de 50khz, un rango sensorial de 0.15 a 10.7 m, con un ángulo de cobertura de 15° a -6 dB.

Cada sonar proporciona la distancia del objeto más cercano dentro de su zona de cobertura. En la Figura 3.2 podemos ver cual es la disposición angular de los sensores en el sistema robótico P3-DX. También hay que decir que, dada la calibración de estos sonares en el robot, tendremos un rango máximo de 5m.

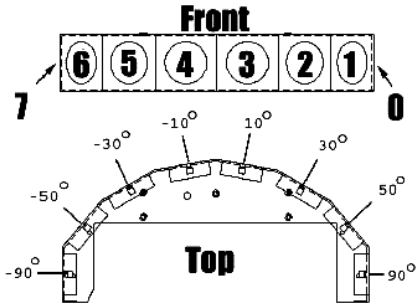


Figura 3.2: Disposición angular de los sonares en el robot P3-DX.

3.2. Entorno y recolección de datos

Para la recolección de datos se coloco al robot en el centro de dos arenas de dimensiones cuadradas, la primera de aproximadamente 3.5 m y la segunda de aproximadamente 5

m las dos con obstáculos de diferentes formas (rectangulares y circulares), tamaños variando de 30 a 60 cm y altura suficiente para estar sobre el horizonte y poder ser procesados por el sistema estéreo de manera adecuada (Figura 3.3).

Figura 3.3: Vista de la arena de recolección de datos.

El robot tiene un diámetro de 38 cm y esta equipado con un sistema de visión estéreo constituido por dos cámaras, el cual toma un par de imágenes en colores o blanco y negro, y un sistema de sensores táctiles simulados por el sistema de sonares. El robot se mueve en línea recta hacia adelante en la arena, tomando fotos del entorno. Cada 15 cm se toma una foto. En la Figura 3.4 podemos ver un ejemplo de imágenes tomadas en la recolección de datos.

Figura 3.4: Ejemplo de imágenes tomadas con el par estéreo en la recolección de datos.

A todos los obstáculos se les ha dotado de una textura, que es un patrón de áreas negras y blancas, para lograr que en el calculo de la imagen de disparidad sean procesados correctamente, sobre todo debido al operador de interés que se menciona en la sección 2.1.5.

Los obstáculos se situaron aleatoriamente en la periferia de la arena. La distancia mínima permitida de cualquier obstáculo al robot estuvo determinada por el horopter capaz de procesar el sistema de visión estéreo.

El horopter se calculó de acuerdo a la ecuación 2.13 y utilizando los parámetros que arrojó la calibración del sistema estéreo:

$$b = 91.68mm$$

$$f = 3.38mm$$

$$pixel = 0.012mm$$

El cálculo se hizo a manera de que los objetos cercanos, alrededor de 34 cm, pudieran ser detectados sin comprometer demasiado el tamaño efectivo de la imagen de disparidad de salida del sistema estéreo. Por lo tanto siendo $z_{min} = 340$ mm para obtener la disparidad en píxeles tenemos lo siguiente:

$$\frac{91.68 \times 3.38}{0.012 \times d + X_{off}} = 340 \Rightarrow d + X_{off} = \frac{91.68 \times 3.38}{0.012 \times 340} = 75.95 \approx 76$$

lo que nos da 64 píxeles de disparidad máxima y 12 píxeles de X-offset. Con esto podemos calcular z_{max} cuando la disparidad es igual a 0:

$$z_{max} = \frac{91.68 \times 3.38}{0.012 \times 12} = 2151.93$$

con lo que queda establecido el horopter a $[340, 2152]$ en milímetros.

Dado que la distancia más cercana debe ser de 34 cm entonces establecimos el umbral de los sonares para detectar una colisión a 44 cm a fin de que si el último movimiento antes de detectar una colisión se da en los 45 cm entonces después de ejecutar el comando motriz y detectar la colisión aun se pueda procesar adecuadamente la imagen de disparidad sin salir del horopter. Es también necesario mencionar que los sonares utilizados para detectar las colisiones son los cuatro sonares de la parte central del sistema de sonares, esto es debido al HFOV de la cámara como se aprecia en la Figura 3.5.

3.3. Diseño del sistema

La información visual es preprocesada antes de ser usada en el sistema. Originalmente, la imágenes que vienen del sistema estéreo tienen un tamaño de 320×240 píxeles, el procedimiento de procesamiento de las imágenes se resume a continuación.

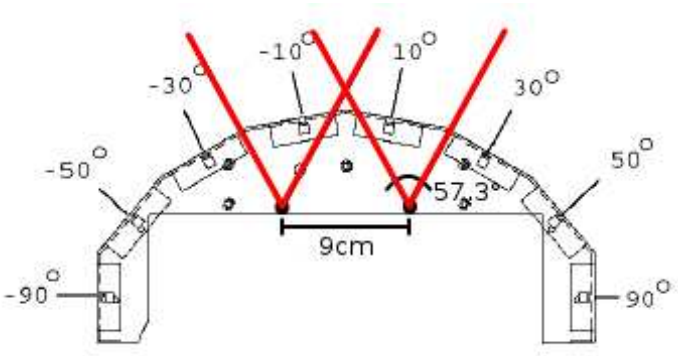


Figura 3.5: Campo de visión de las cámaras y posicionamiento de los sonares.

1. Adquisición de las imágenes a través de la cámara estéreo previamente calibrada y rectificada.

2. Cálculo de la disparidad, con 64 píxeles de disparidad máxima, 12 píxeles de X-offset y una ventana de correlación de 19×19. Esta combinación de parámetros nos da un intervalo de valores de disparidad de 0 - 1024 ya que se cuenta con la interpolación de sub-píxeles a 1/16, es decir, $64 \times 16 = 1024$. En la Figura 3.6 podemos ver la imagen de disparidad generada a partir de la entrada visual mostrada en la Figura 3.4.

Figura 3.6: Ejemplo de imagen de disparidad.

3. Extracción de la región de interés (RI) del mapa de disparidad. Esta RI es una ventana de 228×6. La localización esta gobernada de acuerdo a diversos factores. En la dimensión vertical, se situó el limite superior en la linea 152 de manera que los objetos a más de 2.15 m, que es z_{max}, no sean visibles puesto que no pueden ser procesados

correctamente ya que están fuera del horopter, y el límite inferior vertical se situó en la linea 157 para que las líneas del suelo no interfieran con la información relevante de obstáculos reales. En la dimensión horizontal se tiene que tomar en cuenta que al procesar el mapa de disparidad se pierden secciones verticales en la imagen de salida, dicha perdida está de acuerdo a la disparidad máxima, el X-offset, el tamaño de la ventana de correlación y en menor medida también a la rectificación de la distorsión radial, en nuestro caso aparentemente se pierde más del lado izquierdo, sin embargo la perdida es igual a ambos lados. Esto nos deja una zona efectiva de 228 píxeles en el intervalo que va de la columna 89 a la 316.

4. A continuación se ensambla el vector que representa la máxima disparidad de cada columna (Vector de Máxima Disparidad, VMD) de la imagen en la RI. Es decir que en cada columna se buscará el píxel de mayor disparidad y se colocará en el vector, este vector representa los objetos más cercanos sobre los 57.3 grados de visión de la cámara.

5. Para finalizar se utiliza un filtro de suavizado gaussiano, el cual se utiliza tanto para eliminar ruido como para, de manera más importante, proveer al modelo directo una señal más adecuada para el aprendizaje (*apéndice A*). En este filtro se utiliza una ventana de 5 píxeles.

Este procedimiento se muestra esquemáticamente en la Figura 3.7.

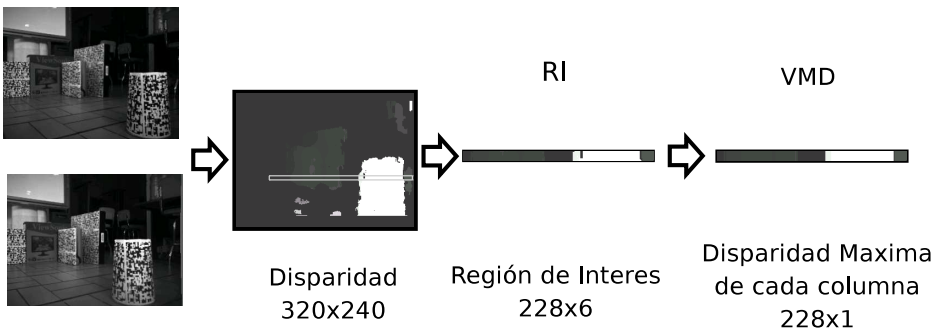


Figura 3.7: Etapas del preprocesamiento de la entrada visual.

Para el entrenamiento se generó un vector de 228 valores representando el valor de los parachoques, en el cual todos los valores valían 1 o 0 dependiendo si se está detectando una

colisión o no respectivamente utilizando solo los dos sonares centrales, ya que prácticamente solo se puede encontrar una correlación de estos con la información disponible en la imagen de disparidad. El umbral aplicado para determinar el valor binario se fijo al valor de 440 que representa la distancia de 44 cm en los sonares.

Como último paso para ajustar los datos para el entrenamiento se le aplicó una normalización al VMD para tener valores en el intervalo [0,1].

Se utilizó un modelo directo para conseguir que el robot hiciera la asociación de la información sensorial con su propia noción de distancia, además que proveyó al robot la capacidad de predecir colisiones. La implementación se muestra en la Figura 3.8 donde D_t es la disparidad en el tiempo t, M_t es el comando motriz al tiempo t, que se considera constante (hacia adelante), D_{t+1} representa la disparidad predicha para el tiempo siguiente $(t+1)$ y finalmente B_{t+1} es el valor predicho de los sensores táctiles para el tiempo siguiente $(t+1)$.

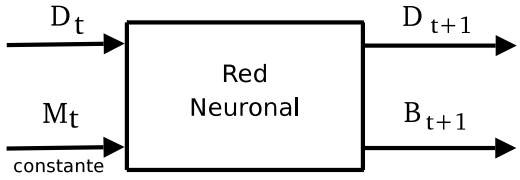


Figura 3.8: Implementación del Modelo Directo.

El sistema realiza una predicción simétrica local. Esto significa que cada modelo directo toma su entrada de una sección de 16 píxeles del VMD y predice los cuatro píxeles centrales de dicha sección así como 4 píxeles del vector del parachoques. En los bordes del VMD no es posible recorrer la sección de entrada, entonces dicha sección es la misma para la predicción de los píxeles en los bordes a manera de utilizar la información disponible (Figura 3.9).

El sistema global consta de 57 redes perceptrón multicapa, esto debido a que el tamaño del VMD es de 228 píxeles y vamos a predecir 4 píxeles por cada red.

El sistema local, es decir, cada una de las redes perceptrón, es una red completamente conectada (significa que cada unidad de una capa se conecta con cada unidad de la capa siguiente) entrenada con el algoritmo de aprendizaje RPROP (*sección 2.3.2*). La estructura de cada red es de 16 unidades en la capa de entrada, 2 capas ocultas cada una

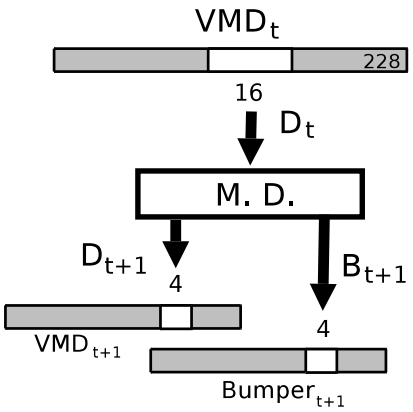


Figura 3.9: Modelado del procesamiento del VMD a través del modelo directo.

con 20 unidades y 8 unidades en la capa de salida, cuatro para la predicción del VMD al tiempo siguiente y cuatro para la predicción del vector de los parachoques.

Se prepararon 2259 patrones de entrenamiento entre los cuales había patrones con colisiones y sin colisión.

Capítulo 4

Experimentos

Se entrenaron las redes por 5089 ciclos de entrenamiento off-line, las pruebas se realizaron con patrones que no fueron utilizados para el entrenamiento. Se deseó que la red pudiera realizar dos tipos de predicciones. La primera de ellas es la *predicción de un paso (PUP)*, es decir, dado el valor de D_t predecir los valores de D_{t+1} y B_{t+1}. Esta es la salida de la red neuronal. La segunda es la *predicción de largo plazo (PLP)* la cual consiste de usar la salida visual predicha como entrada al sistema nuevamente. Esta predicción es como una simulación interna del robot.

En la PUP los valores reportados en la activación de los parachoques no son completamente binarios, sino que la activación esta cercana a 0 cuando no se detecta colisión y cuando hay una colisión esta cerca a 1. Más importante, es que las redes que incrementan su activación hasta casi 1 son aquellas que se encuentran en el lado del robot donde el obstáculo está próximo o aquellas donde ocurren los cambios más significativos en el campo visual. Se implementó un umbral para la PUP para indicar una colisión, si más de 4 neuronas muestran una activación superior a 0.95 entonces se reporta una colisión.

Para que el sistema dispare la PLP se definió de igual manera un umbral. Cuando mas de 35 neuronas del VMD predicho presentan una activación mayor a 0.45 en el PUP, entonces comienza la simulación interna para el resto de la trayectoria. Se probó está configuración en varias trayectorias y siempre que hubiera una futura colisión se disparó la PLP, de manera similar cuando no había colisión no se activó la PLP.

En la Figura 4.1 podemos ver los mapas de disparidad de las primeras imágenes en dos distintas trayectorias típicas utilizadas en la evaluación del sistema.

La salida visual de la primera de ellas se muestra en la Figura 4.2, en la figura

Figura 4.1: Imagen de disparidad de dos trayectorias de prueba.

izquierda vemos los VMDs reales esperados de la predicción, en la central vemos la predicción de los VMDs de un paso, y en la derecha la predicción de largo plazo. Se muestra una trayectoria de 11 pasos con una colisión en el lado derecho del robot, como se puede ver el sistema dispara la PLP 3 pasos antes de que suceda la colisión. Esto significa alrededor de 45 cm antes de que se presente la colisión, además que cuando se presenta la colisión el robot se encuentra a por lo menos a 34 cm, es decir que en realidad se detecta la colisión casi a 80 cm antes de que ocurra, esto es más que la propia dimensión del robot que como ya se había mencionado es de 38 cm.

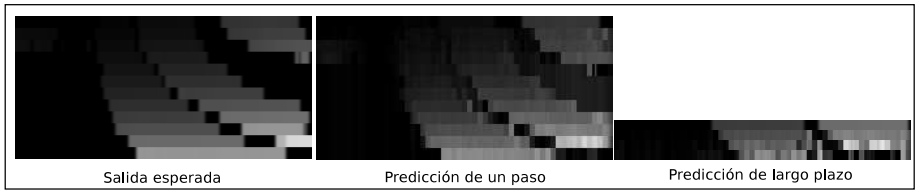


Figura 4.2: Evaluación del sistema con colisión en el lado derecho.

En la Figura 4.3 podemos apreciar la activación predicha de los parachoques, podemos observar como las neuronas de salida presentan mayor activación son las situadas en el lado del robot donde la colisión se va a producir.

La salida del segundo ejemplo se muestra en la Figura 4.4 el cual presenta el comportamiento del sistema cuando se presenta una colisión en el lado izquierdo. En este caso se presenta una trayectoria de 4 pasos. La PLP se dispara 3 pasos antes de que suceda la colisión.

En la Figura 4.5 se muestra la activación predicha de los parachoques, en está

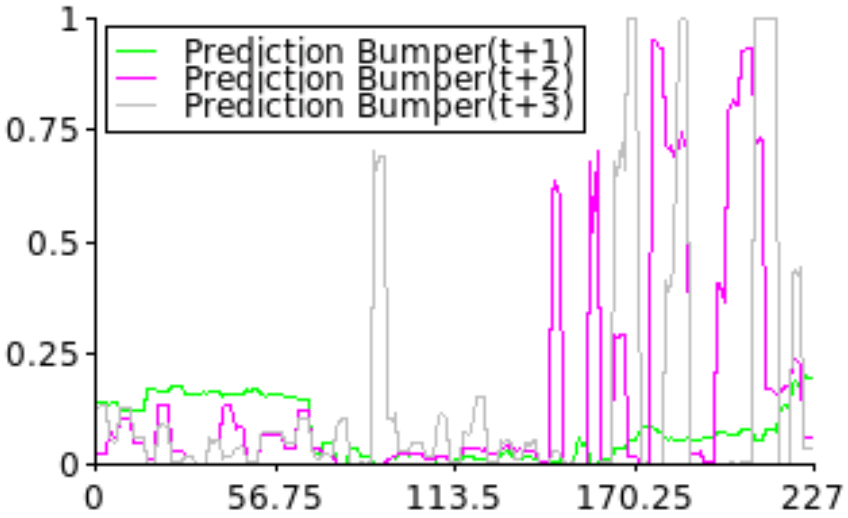


Figura 4.3: Activación predicha de los parachoques.

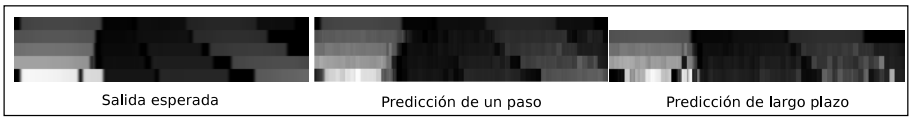


Figura 4.4: Evaluación del sistema con colisión en el lado izquierdo.

ocasión las neuronas que presentan mayor activación son las localizadas en el lado izquierdo.

Es preciso hacer notar que en la primer trayectoria mostrada, el obstáculo contra el que se detecta la colisión (el coloreado en tono gris oscuro en la Figura 4.1 izquierda) en el ultimo paso ya no aparece en el VMD puesto que está fuera del campo de visión del sistema, sin embargo, existe una colisión real, mas aun, el sistema, al tener la información en el paso anterior detecta la colisión correctamente solo debido a la activación de las neuronas de salida del parachoques.

Medición del error

La métrica, *suma de los errores al cuadrado (SSE)*, se aplica a los datos visuales para encontrar una medida objetiva del error, es decir:

$$SSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2$$

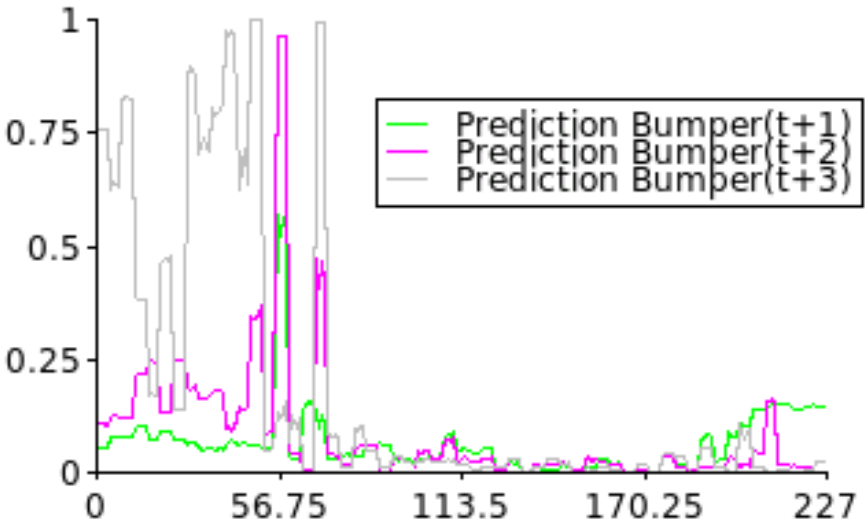


Figura 4.5: Activación predicha de los parachoques.

donde x es el dato real y $\hat{x}$ es el dato predicho.

En la figura 4.6 se muestran 7 diferentes trayectorias durante 10 pasos de tiempo de la Predicción de Largo Plazo (PLP).

Como se puede observar el error es acumulativo, a mayor número de cadenas de predicción más error existe, sin embargo considerando el hecho de que para 10 cadenas de predicción se tiene una distancia de cobertura de alrededor de 1.85 m entonces se considera un error aceptable.

4.1. Implementación

Una vez entrenado el modelo directo se implemento en el robot. Los datos provenientes de las cámaras son preprocesados y alimentados a la red mientras el robot se mueve.

Se probó el modelo en dos experimentos de navegación. En ambos se planteo la tarea de evasión de obstáculos.

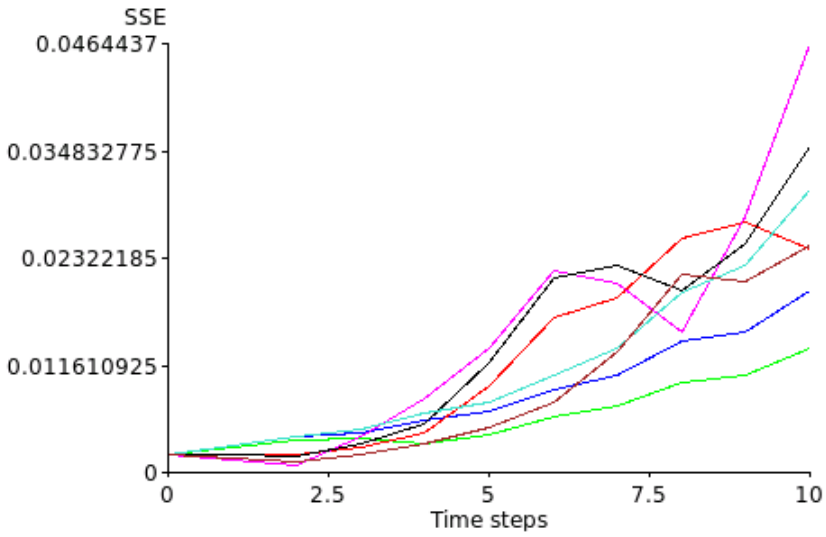


Figura 4.6: Error SSE de 7 trayectorias tipicas durante 10 pasos de tiempo de la PLP.

4.1.1. Experimento 1

En el primer experimento se planeo que el robot se movíera hacia el frente mientras iba evitando los obstáculos que encontraba a su paso. Para llevar a cabo esta tarea el robot utilizó el modelo directo previamente programado. El modelo esquemático lo podemos apreciar en la Figura 4.7.

En este esquema, como podemos ver, en cada paso se genera una entrada sensorial, que posteriormente es procesada para obtener el VMD en base a ello el robot realiza una PUP, si es que el valor del VMD predicho alcanza el umbral de activación previamente especificado se dispara la PLP, en la PLP se verifica si en 4 pasos o menos se presentaria una colisión, si no se detecta una colisión entonces se continua con el sistema motriz ejecutando el paso hacia adelante, si es que si detecta una colisión entonces se revisa de que lado proviene la colisión y se gira al lado contrario para seguir con el sistema motriz y el comando girar.

El tamaño del paso se fijo a 15 cm. El umbral para determinar el disparo de la PLP es determinado por la activación a un valor de 0.45 o mayor de 35 o más neuronas del VMD predicho. El umbral para detectar una colisión se fijo a un valor de activación de 0.95 de más de 4 neuronas de los parachoques predichos.

Es importante recalcar que el giro esta determinado únicamente por la activación de las neuronas del parachoques predicho en la PLP. El giro se estableció a 35 grados, para

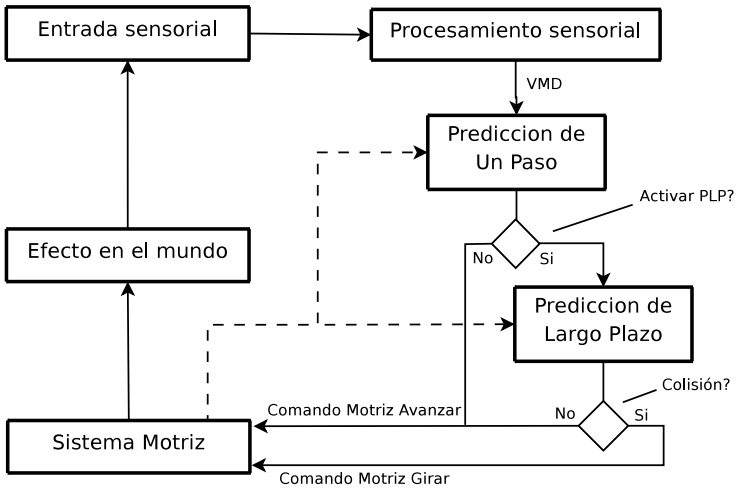


Figura 4.7: Modelo esquemático del primer experimento.

determinar la dirección se identificó la zona con mayor número de neuronas táctiles activas a más de 0.95. La separación de las zonas izquierda y derecha en el VMD es simétrica, por lo tanto la región izquierda se definió en el intervalo [0,114] píxeles y de [115,228] se consideró la región derecha.

En este experimento se podía dar el caso de que al realizar la PUP se detectara una colisión, esto podría suceder por ejemplo debido a que después de un giro un obstáculo quedara muy próximo al robot. En dichos casos el robot retrocede por un paso.

La Figura 4.8 muestra la interfaz del sistema en funcionamiento.

Se definió un número máximo de 25 pasos para cada trayectoria, esto a manera de controlar el tiempo de prueba.

Se probó al robot colocándolo en el extremo de una arena en la cual se colocaron diversos obstáculos de formas rectangulares y circulares, así como tamaños variados de 30 a 60 cm y altura suficiente para ser detectados por el sistema de visión estéreo y poder ser procesados de manera adecuada. En la Figura 4.9 podemos ver el posicionamiento del robot en la arena en una trayectoria típica de prueba.

En la Figura 4.10 se muestran seis instantes en la trayectoria típica de prueba mostrada en la Figura 4.9.

- Al comenzar la trayectoria el robot avanza un paso activa la PLP detecta una colisión en 2 pasos y gira a la izquierda (Figura 4.10.(a)).

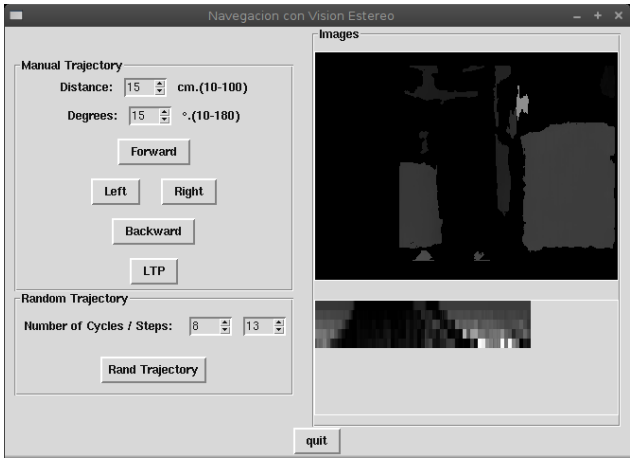


Figura 4.8: Interfaz del sistema.

- Al hacer el giro queda muy cerca de la pared de cuadros y detecta una colisión en la PUP por lo que retrocede un paso (Figura 4.10.(b)).
- Después el robot gira a la derecha y al encontrar el obstáculo vuelve a girar a la derecha (Figura 4.10.(c)).
- A continuación el robot avanza hasta disparar la PLP 3 pasos antes de llegar al obstáculo por lo cual gira a la izquierda (Figura 4.10.(d)).
- En seguida el robot avanza varios pasos hasta llegar al fondo (Figura 4.10.(e)).
- Para finalizar detecta una colisión en 3 pasos y gira a la derecha (Figura 4.10.(f)) con lo cual concluye la trayectoria de 25 pasos.

El sistema se probó en 21 trayectorias diferentes de 25 pasos, obteniendo muy buenos resultados, puesto que la PLP en el 98 % de los casos se activo adecuadamente, cuando no se llego a activar fue debido al problema de la reducción del mapa de disparidad en el lado izquierdo. Sin embargo no obstante nunca se activo en falso la PLP. Además que la PLP detecto las colisiones a 3 pasos en promedio, lo que representa una distancia suficiente para evitar los obstáculos dado el tamaño del robot. También cabe mencionar que el giro se dio de manera adecuada de acuerdo a lo sugerido por la activación de los parachoques predichos por la PLP.

Figura 4.9: Ejemplo típico de prueba en el experimento 1.

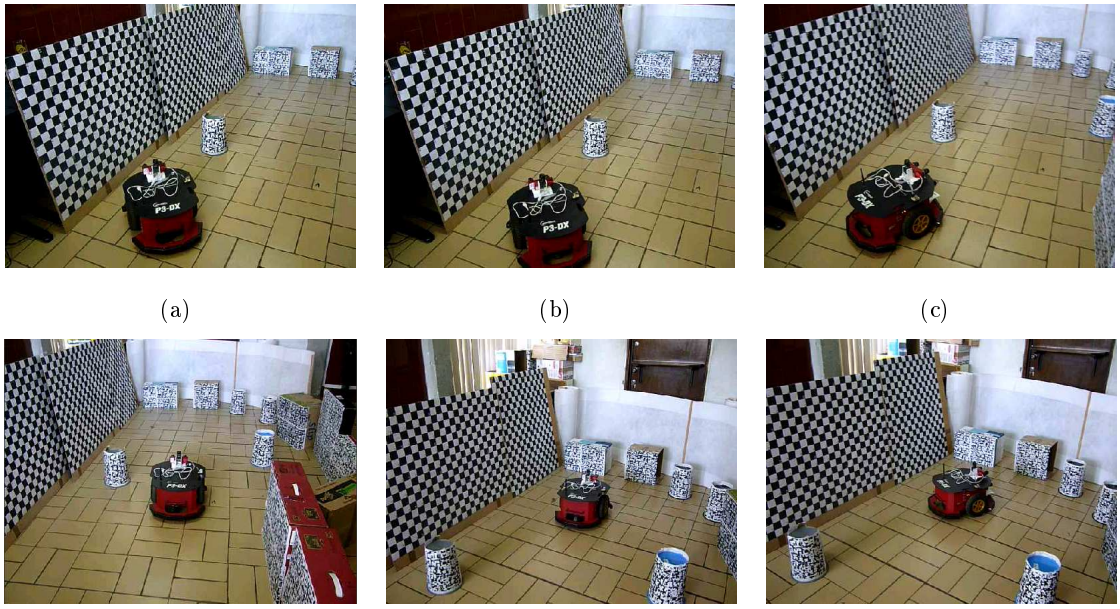
4.1.2. Experimento 2

Para el segundo experimento se planeó que el robot buscara la salida encontrándose rodeado de obstáculos. Para llevar a cabo la tarea se pensó que el robot girara sobre su eje para explorar el entorno y decidiera el camino más adecuado para comenzar a avanzar evitando los obstáculos que encuentre a su paso. De igual manera se utilizó el modelo directo previamente entrenado. Se determinaron 3 comportamientos para el robot los cuales se muestran esquemáticamente en la Figura 4.11.

Los comportamientos definidos son:

1. **Exploración del entorno.** En este comportamiento el robot rota sobre su propio eje los 360 grados en pasos de 10 grados, en cada paso hace una PLP y almacena la información en un vector para formar una representación del entorno. En la Figura 4.12 vemos un esquema del comportamiento.

2. **Posicionamiento.** En este comportamiento el robot rota sobre su propio eje para posicionarse en la orientación más *prometedora*. Dicha orientación depende de la búsqueda del valor que se ajuste a dos criterios en el vector entregado por el comportamiento anterior. Como la posición de este valor en el vector tiene una relación directa con los grados desde la posición inicial de la exploración entonces es trivial ajustar la orientación. El primero de los criterios selecciona el valor que en el vector tenga la mayor diferencia con respecto a las posiciones contiguas. El segundo criterio solo en caso de que el criterio anterior nos de más de un dato y se trata de seleccionar el dato con



(a) (b) (c)

(d) (e) (f)

Figura 4.10: Trayectoria de prueba típica del experimento 1.

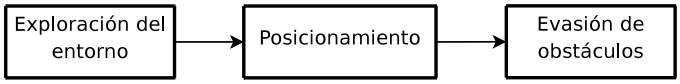


Figura 4.11: Diagrama de los comportamientos utilizados en el segundo experimento.

mayor valor.

3. **Evasión de obstáculos.** Este comportamiento lo que hace es mover el robot hacia adelante mientras evita los obstáculos, esta completamente inspirado en el experimento 1. Es decir que los valores de los umbrales son los mismos que para el experimento 1, así como el giro cuando se detecta una colisión. El diagrama esquemático es el mismo que el mostrado en la Figura 4.7.

La interfaz del sistema es la misma que para el experimento 1.

Se le definió un número máximo de 20 pasos para el comportamiento de evasión de obstáculos.

Se probó al robot situado en una arena en la cual se colocaron alrededor de él diversos obstáculos de formas y tamaños similares a los del experimento anterior. En todos

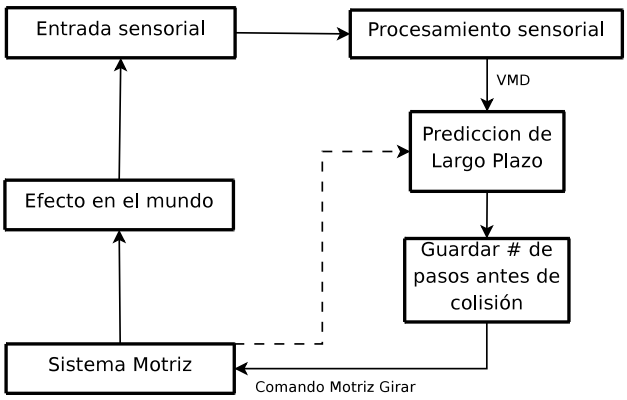


Figura 4.12: Exploración del entorno 1.

los casos se dejo un espacio, de tamaño suficiente para que el robot pueda pasar, entre algún par de obstáculos a manera de salida del arreglo de obstáculos, este espacio es el que el robot debe identificar para determinar que realmente el robot entiende la distancia en función de sus características, en este caso su tamaño propio.

- En la Figura 4.13 podemos ver el posicionamiento del robot en la arena en una de las trayectorias de prueba.

Figura 4.13: Posicionamiento del robot en la arena.

- En la Figura 4.14 se muestran dos instantes en el comportamiento de exploración del entorno.

(a) (b)

Figura 4.14: Exploración del entorno 2.

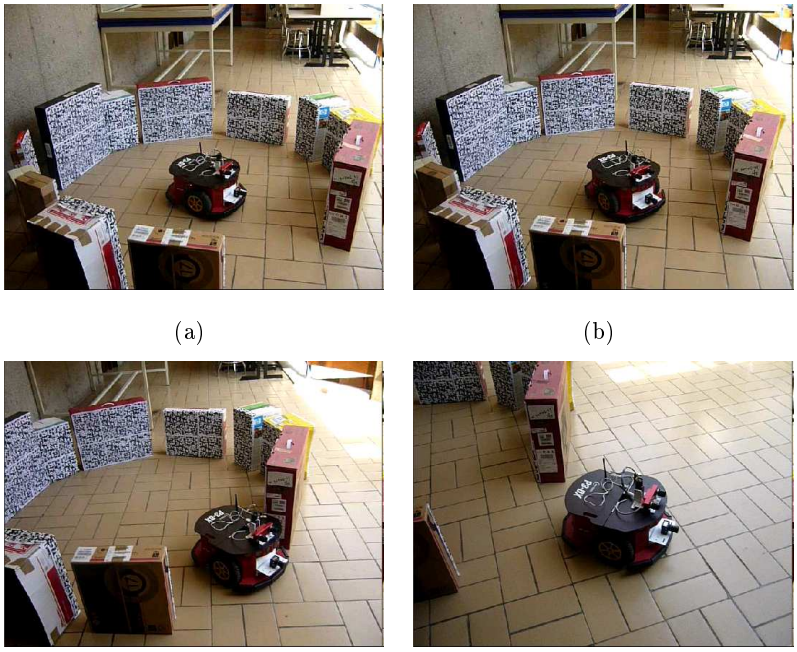
- Posteriormente el robot entro en el comportamiento de posicionamiento y se oriento hacia el espacio libre del arreglo de obstáculos, como se muestra en la Figura 4.15.

Figura 4.15: Posicionamiento del robot hacia el espacio libre.

- Después se comienza con la evasión de obstáculos, en la Figura 4.16 podemos ver 4 instantes en la evasión de obstáculos.

Como se ve en la Figura 4.16.(a) el robot en el posicionamiento no logra completamente a orientarse a la salida debido al patinaje en las ruedas, este es un error acumulativo al hacer la exploración del entorno, sin embargo como se en la Figura 4.16.(b) el robot en el último comportamiento corrige ese pequeño error. En las dos figuras restantes podemos ver como el robot logra salir del arreglo de obstáculos.

Se probó el mismo experimento en diferentes entornos en donde distinguir la salida es más complejo. En la Figura 4.17 podemos ver otra secuencia de ejecución. Las Figuras 4.17.(a) y (b) corresponden a la exploración del entorno, el posicionamiento lo podemos ver en la Figura 4.17.(c) y en las figuras restantes 4.17.(d) y (e) se encuentra la evasión de



(a) (b)

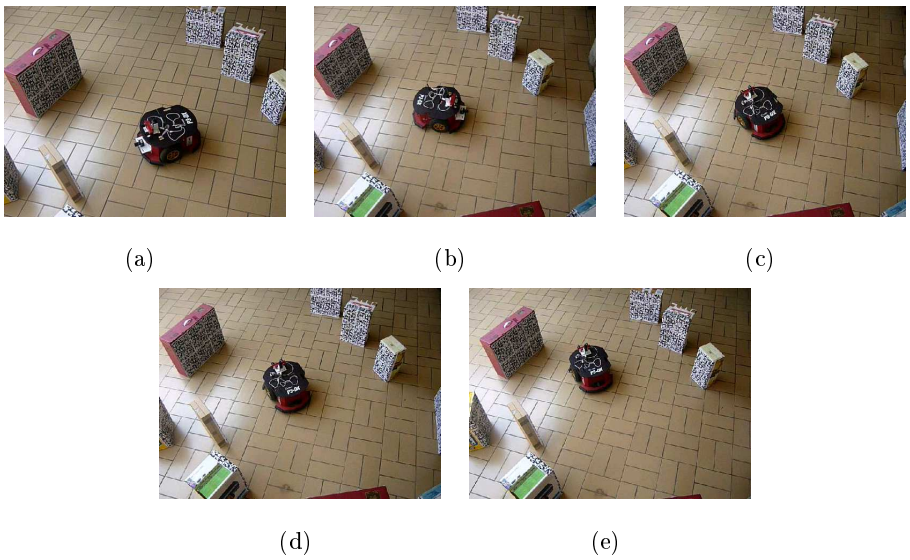
(c) (d)

Figura 4.16: Evasión de obstáculos.

obstáculos.

Se muestra por último otro entorno en la Figura 4.18. Como podemos observar en las Figuras 4.18.(a) y (b) el robot distingue correctamente la salida, aun cuando hay un obstáculo un poco más al fondo. Por último el robot evade los obstáculos de manera adecuada (Figuras 4.18.(c), (d), (e) y (f)).

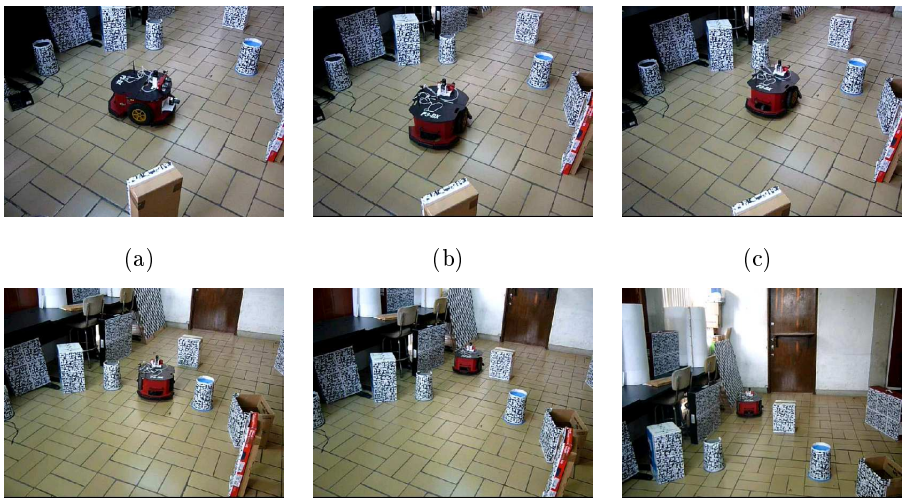
El sistema se probó en varios entornos diferentes obteniendo buenos resultados, puesto que el robot encontró en todos los casos la salida, en el 90 % de los casos el robot aproximo la orientación de la salida de manera adecuada pero no exactamente debido a el error que el patinaje de las llantas produce acumulativamente en la fase de exploración.



(a) (b) (c)

(d) (e)

Figura 4.17: Comportamiento del robot en la arena con obstáculos.



(a) (b) (c)

(d) (e) (f)

Figura 4.18: Comportamiento del robot en la arena con obstáculos.

Capítulo 5

Conclusiones y trabajo futuro

En el desarrollo y los experimentos presentados en este trabajo de tesis se muestra que al robot le es útil el mecanismo que provee el modelo directo para realizar una asociación de la información multimodal a la distancia relativa a su propias características.

El robot mediante el sistema diseñado es capaz de anticipar las colisiones, así como distinguir un espacio por donde puede pasar de por donde no puede hacerlo de manera autónoma. Estos resultados señalan que internamente se da una asociación de características intrínsecas del robot y capacidades de interacción en el ambiente.

Aun mas, la predicción puede proveer al robot con estrategias útiles para la exploración y la navegación. El agente presentado aquí es capaz de predecir las consecuencias de sus propias acciones y al mismo tiempo tomar previsiones para las acciones futuras.

También hay que mencionar que el sistema aprende a asociar los estímulos visuales y táctiles y forma con ello una representación multimodal, esta representación multimodal es la que le sirve al robot para entender el entorno y se forma mediante la interacción con el ambiente.

Se observa de igual manera que los valores de activación de los parachoques proveen al sistema con más información de la que se esperaba al entrenar, puesto que su activación nos dice de que lado se presentara la colisión. Es importante señalar que la decisión del giro tomada por el agente se basa únicamente en los valores de activación del parachoques, no en los estímulos visuales.

Como trabajo futuro se proponen los siguientes tópicos:

Comandos motrices no constantes. En el presente trabajo se considero solamente el comando motriz avanzar por una distancia fija, se piensa que el uso de comandos

motrices variados, como son los giros y avance en distintos pasos, puede proveer mayor información al sistema y por tanto tener una gama más amplia de posibilidades en la toma de decisiones.

Dinámica de la red. Al investigar la dinámica interna que rige el comportamiento de la red neuronal se espera encontrar patrones de activación interesantes como pueden ser la asociación de la activación de ciertas unidades a los cambios de intensidad, así como la integración sensorial multimodal.

Entornos dinámicos. Creemos que el desarrollo del sistema para entornos dinámicos cambiantes en tiempo real puede hacer al sistema más general, puesto que en la realidad los entornos no son estáticos. Además que se piensa que, dado que el entorno requeriría procesamiento en tiempo real esto llevaría al desarrollo de un algoritmo de detección de características relevantes del mapa de disparidad para concentrar el procesamiento en dichas características.

Diferentes algoritmos de obtención del mapa. Las herramientas y los algoritmos para obtener el mapa de disparidad son numerosos y variados, se piensa que el uso de los métodos basados en características puede arrojar resultados interesantes en cuanto que se tendría a priori el reconocimiento de objetos y contornos. También se piensa en la posibilidad de utilizar un sistema estéreo con movimiento horizontal (*pan*) para crear un segundo y tercer mapa a los costados, y unificar los tres mapas para ampliar la región efectiva resultante del mapa de disparidad, así como para aumentar la confianza gracias a la redundancia en la información.

Vergencia de las cámaras. Una linea de investigación puede desprenderse al considerar cámaras no estáticas si no con la propiedad de girar y tener un sistema con vergencia y seguimiento visual, esta podría ser una propuesta interesante sobre todo en un entorno dinámico. De igual manera esta misma linea puede llevar a desarrollar investigación en el campo de la atención visual y la visión activa.

Bibliografía

[1] R. Möller and W. Schenk, "Bootstrapping cognition from behavior— a computerized thought experiment," *Cognitive Science*, vol. 32, pp. 504–54, April 2008.

[2] M. Wilson and M. Wilson, "Six views of embodied cognition," *Psychonomic Bulletin and Review*, vol. 9, pp. 625–636, 2002.

[3] B. Raducanu and J. Vitriá, "Learning to learn: From smart machines to intelligent machines," *Pattern Recogn. Lett.*, vol. 29, no. 8, pp. 1024–1032, 2008.

[4] B. Lara, J. M. Rendon, and M. A. Capistran, "Prediction of multi-modal sensory situations a forward model approach," in *Proc. of the 4th IEEE Latin America Robotics Symposium*, vol. 1, 2007.

[5] S. Harnad, "The symbol grounding problem," *Physica D*, vol. 42, pp. 335–346, 1990.

[6] R. A. Brooks, "Elephants don't play chess," *Robotics and Autonomous Systems*, vol. 6, pp. 3–15, 1990.

[7] B. M. Collins and A. L. Kornhauser, "Stereo vision for obstacle detection in autonomous navigation," tech. rep., Princeton University, May 24 2006.

[8] A. Murarka and B. Kuipers, "A stereo vision based mapping algorithm for detecting inclines, drop-offs, and obstacles for safe local navigation," in *Intelligent Robots and Systems, IEEE/RSJ International Conference on*, pp. 1646 –1653, 10-15 2009.

[9] D. Murray and J. J. Little, "Using real-time stereo vision for mobile robot navigation," *Auton. Robots*, vol. 8, no. 2, pp. 161–171, 2000.

[10] J. Bruce and M. Veloso, "Real-time randomized path planning for robot navigation," in *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, vol. 3, pp. 2383–2388, December 2002.

[11] D. Hähnel, R. Triebel, W. Burgard, and S. Thrun, "Map building with mobile robots in dynamic environments," in *In In Proc. of the IEEE International Conference on Robotics and Automation (ICRA*, 2003.

[12] U.Ñehmzow and B. McGonigle, "Robot navigation by light," European Conference on Artificial Life, ECAL, May 1993.

[13] M. Zamora, H. Martinez, A. Skarmeta, and L. Tomas-Balibrea, "Navegacion planificada de un robot movil en entornos interiores desconocidos," 2000.

[14] M. Batalin, M. Hattig, and G. S. Sukhatme, "Mobile robot navigation using a sensor network," in *In IEEE International Conference on Robotics and Automation*, pp. 636–642, 2003.

[15] A. H. A. Hasan, R. A. Hamzah, and M. H. Johar, "Region of interest in disparity mapping for navigation of stereo vision autonomous guided vehicle," *Computer Technology and Development, International Conference on*, vol. 1, pp. 98–102, 2010.

[16] M. Kawato, "Internal models for motor control and trajectory planning," *Current Opinion in Neurobiology*, vol. 9, no. 6, pp. 718–727, 1999.

[17] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521623049, 2000.

[18] T. Moons, "A guided tour through multiview relations," in *SMILE'98: Proceedings of the European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*, (London, UK), pp. 304–346, Springer-Verlag, 1998.

[19] R. Y. Tsai, "Radiometry," ch. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses, pp. 221–244, , USA: Jones and Bartlett Publishers, Inc., 1992.

[20] K. Konolige, "Small vision system. hardware and implementation," in *In Proc. International Symposium on Robotics Research*, (Hayama, Japan), pp. 111–116, 1997.

[21] J. P. Graffigna, L. E. Romero, and R. Romo, "Evaluación de métodos para la obtención del mapa de disparidad en sistemas de visión estéreo," *Congreso Argentino de Bioingenieriat*, vol. 11, pp. 1–4, 2007.

[22] B. Webb, "Neural mechanisms for prediction: do insects have forward models?," *Trends in Neurosciences*, vol. 27, pp. 278–282, May 2004.

[23] R. Miall and D. Wolpert, "Forward Models for Physiological Motor Control," *Neural Networks*, vol. 9, pp. 1265–1279, November 1996.

[24] D. M. Wolpert, Z. Ghahramani, and M. I. Jordan, "An internal model for sensorimotor integration," *Science*, vol. 269, pp. 1880–1882, 1995.

[25] S. Russell and P.Ñorvig, *Inteligencia Artificial: un enfoque moderno.* Madrid: Prentice Hall, 2004.

[26] R. Rojas, *Neural Networks: a systematic introduction.* Springer-Verlag, 1996.

[27] B. Krose and P. van der Smagt, *An introduction to Neural Networks.* octava edición ed., November 1996.

[28] J. L. Elman, "Finding Structure in Time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[29] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 79, pp. 2554–2558, April 1982.

[30] T. Kohonen, *Self-organization and associative memory.* Springer-Verlag New York, Inc. New York, NY, USA, 1989.

[31] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: the RPROP algorithm," pp. 586–591, 1993.

Apéndice A

Suavizado Gaussiano

El suavizado, o también alisado, es una operación de procesamiento de imágenes simple y frecuentemente usada. Hay varias razones para suavizar, usualmente se hace para reducir el ruido, sin embargo nosotros tenemos un motivo particular que es el adaptar la señal de entrada de la red neuronal para que el aprendizaje sea más fácil. Entre los diferentes filtros de suavizado, el *filtro Gaussiano*, es probablemente el más útil aunque no el más rápido.

El filtrado Gaussiano se hace convolucionando cada punto de la matriz de entrada con un kernel Gaussiano de $n_x \times n_y$. En términos generales lo que hace el kernel es calcular la media ponderada, donde los pesos toman la forma de una campana de Gauss.

La campana de Gauss esta determinada por la siguiente ecuación:

$$f(x) = e^{-(x^2+y^2)/s^2}$$

donde x y y son las coordenadas en la matriz de convolución, s^2 es la varianza (Figura A.1).

La varianza, s^2, indica el nivel de suavizado.

- Varianza grande: campana más ancha, más suavizado.
- Varianza pequeña: campana más estrecha, menos suavizado.

El valor de la varianza puede ser determinado en función del tamaño de la ventana de convolución mediante las siguientes ecuaciones:

$$s_x = \left(\frac{n_x}{2} - 1\right) * 0.3 + 0.8$$

$$s_y = \left(\frac{n_y}{2} - 1\right) * 0.3 + 0.8$$

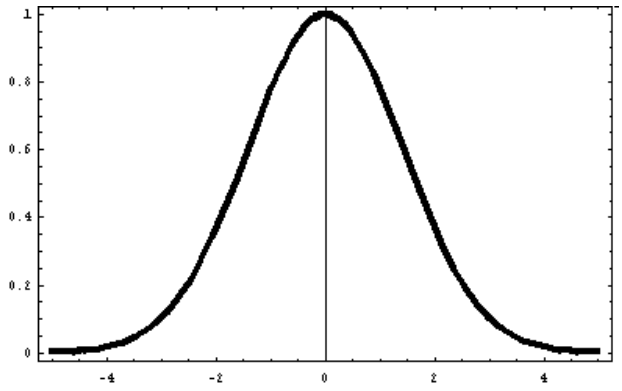


Figura A.1: Campana de Gauss.

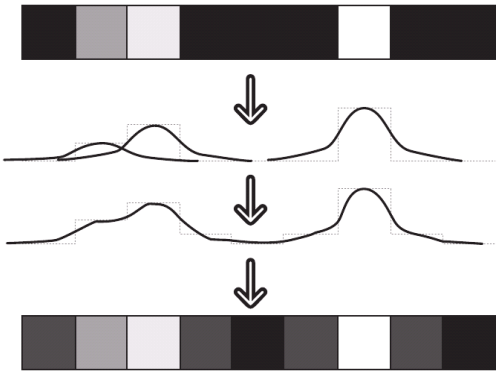


Figura A.2: Suavizado Gaussiano sobre un arreglo de píxeles de 1D.

En la Figura A.2 podemos apreciar el filtrado sobre un arreglo en 1D.

En nuestro caso dado que el filtro se aplica sobre el VMD se utiliza una ventana de convolución de 1D de 5×1. En la Figura A.3 se aprecia el resultado de aplicar el filtro sobre una matriz de VMD's, en la que cada fila es un VMD.

En nuestro caso particular el suavizado se aplica, como se menciono, para que señal del VMD pueda ser aproximada de mejor manera por la red neuronal. En la Figura A.4 vemos el VMD antes y después del suavizado, se han añadido también la representación de las neuronas correspondientes.

Como se puede observar tenemos cambios muy abruptos en el VMD original, lo cual implica que a las neuronas topológicamente contiguas les corresponda una señal completamente diferente, ya sea a la entrada o a la salida que se desea aproximar. En la imagen

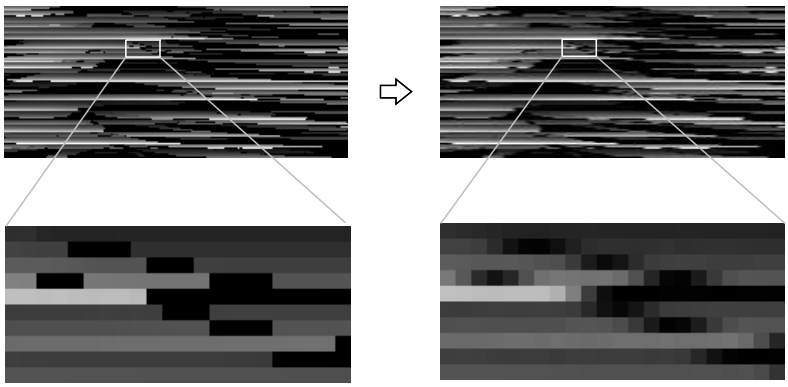


Figura A.3: Suavizado gaussiano sobre una matriz de VMD's.

percibida del mundo las regiones diferenciadas por este cambio de profundidad son espacialmente contiguas. En la red, este cambio abrupto afecta el aprendizaje puesto que la red es un todo y las neuronas contiguas guardan una estrecha relación debido a la dinámica interna. La aplicación de un filtro o suavizado hace que las neuronas de entrada y salida trabajen con datos mas parecidos a lo que existe en la naturaleza. Los efectos de este filtro se pueden ver en el sencillo experimento descrito a continuación.

Se probo un conjunto de patrones de entrenamiento consistente de 617 patrones con y sin suavizado durante 250 ciclos de entrenamiento, en el conjunto sin suavizado el error mínimo que se alcanzo fue de 0.0341701, mientras que en conjunto con suavizado se alcanzo un error mínimo de 0.025466 con una razon de disminución de tiempo de 0.9854. En el entrenamiento se comprobó que el uso del suavizado arrojaba un error más pequeño y un tiempo de convergencia menor.

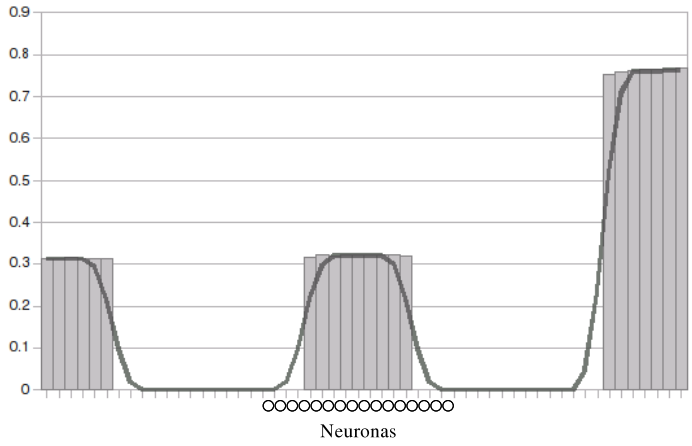


Figura A.4: Representación de un VMD de entrada de una red.

Apéndice B

Software Desarrollado

Se desarrollaron 6 programas que constituyen el sistema de implementación y prueba de código de C++ en el entorno de trabajo de control de versiones vctl y el software ARIA del robot P3-DX.

- ImageCollect v. 1.1: Programa que realiza la recolección de los datos que se utilizarán para el entrenamiento. Genera un conjunto de imágenes en cada paso de la trayectoria de recolección, así como un archivo que contiene los nombres de las imágenes y los valores de los sonares asociados a las imágenes.
- GetVMD v. 0.1: Programa para preparar los datos para el entrenamiento. Es en este programa que se computan los VMD y los vectores del parachoques.
- NetMMR v. 2.2: Programa que entrena la red neuronal utilizando el algoritmo RPROP. En este programa se considera la topología de la red propuesta en este documento de tesis.
- StereoEval v. 1.3: Programa que evalua el desempeño de la red con patrones de prueba.
- StereoTest v. 0.2: Programa de aplicación del sistema para evasión de obstáculos mencionado en el experimento 1.
- StereoTest v. 1.0: Programa de aplicación del sistema para determinar la salida en un arreglo de obstáculos mencionado en el experimento2.