

Robótica y Visión: de Computación y Matemáticas hacia las Ciencias

L. A. Muñoz, A. Espinosa

LI²CoViR*

Facultad de Matemáticas

Universidad Autónoma de Yucatán

Mérida, Yucatán

México

Resumen

Dentro de las actividades del Laboratorio de Investigación en Instrumentación, Control, Visión y Robótica hemos identificado algunos temas de interés común hacia las neurociencias, la bioinformática, el diseño de nanomateriales, la arqueología y la protección al medio ambiente. En cada una de ellas encontramos sistemas complejos que al ser observados o simulados, nos ayuda a entender y encontrar las características más sobresalientes, las cuales podemos hacer materia computacional de estudio. Presentaremos algunos resultados que consideramos de interés para la comunidad científica y académica en los cuales la computación y las matemáticas son, tales que herramienta, los instrumentos centrales.

1. Introducción

El desarrollo de investigación científica en áreas sumamente jóvenes, como lo son la robótica y la visión computacional nos remite, por una parte, a la revisión cuidadosa de sus fundamentos teóricos – tanto físicos como matemáticos –, así como a los aspectos tecnológicos más avanzados involucrados en el desarrollo práctico de estas áreas. Dada la juventud de éstas, es bastante común encontrar que hay un núcleo importante de investigadores en robótica y visión computacional, que siendo especialistas en computación o bien aficionados y que, tras un peregrinaje dentro de las ciencias exactas o la ingeniería, se acercan a estos dominios, que se caracterizan por su carácter multidisciplinario. La naturaleza de la investigación en robótica y visión, y las acciones para concretar esta investigación – desarrollar prototipos experimentales, publicar artículos en congresos y en revistas especializadas– ha ido adoptando un carácter excluyente, lo que impide a grupos carentes de medios experimentales profesionales incursionar de manera efectiva en estas áreas. Por estas razones, no es raro que encontremos en laboratorios universitarios y en otros centros de investigación y enseñanza la utilización de equipo genérico, comúnmente obtenido tras desmantelar o modificar equipo diseñado para otros fines. ¿Cuántos investigadores en robótica no han tenido

en algún momento de su carrera un motor eléctrico en sus manos, cuyo origen puede haber sido incierto, pero cuyo destino ha sido, aunque quizás en sueños, el elemento motriz de un robot?. Tanto en el área de robótica como en el de visión, por lo menos a nivel universitario se trabaja con equipo que no podríamos calificar como el perteneciente al estado del arte, principalmente por razones económicas. Esto nos lleva al uso y fomento del uso de arquitecturas abiertas, tanto en hardware como en software, como un principio básico al momento de la implementación de prototipos y equipo de investigación. Pero no es solo el carácter de irreverencia el que lleva a la creación de nuestras propias herramientas, también somos testigos de como algunos laboratorios se encuentran inundados de equipo cerrado cuya mayor aportación es el simple carácter ilustrativo de la temática y no verdaderas herramientas de experimentación. Las áreas de robótica y visión computacional pueden ser caracterizadas por su naturaleza como los componentes motriz y sensorial de un sistema autónomo. Mientras la robótica trata principalmente de la acción, en un manipulador mecánico sumamente flexible, la visión computacional otorga de fuentes más ricas de información a dichos sistemas. Tanto la robótica como la visión son áreas en donde la incertidumbre, además de ser algo inevitable, define el interés más sensato dentro del dominio ingenieril. Tanto aquella caracterizada por el planteamiento del problema en sí, como aquella proveniente de los instrumentos mismos de movimiento y sensado; ésta última magnificada cuando – como fue mencionado en el párrafo anterior– utilizamos dispositivos “off the shelf”. Un paradigma básico que caracteriza el trabajo del laboratorio es la concepción de que tanto la robótica como la visión no son áreas de investigación excluyentes, sino que tanto los problemas de visión como los de robótica son complementarios, en tanto la detección de fuentes de incertidumbre en ambas áreas, pueden ser caracterizadas y entendidas de una mejor manera al considerar un planteamiento común de la problemática. Así pues, tanto la robótica como la visión por computadora forman parte integral no solo de las acciones sino de la materia de estudio de nuestro interés. Son los instrumentos necesarios para el modelado y estudio de sistemas complejos cuyo entendimiento es la razón principal de nuestro trabajo. Nuestros proyectos cumplen 2 tipos de vinculación. Por un lado nos interesa fomentar las actividades de investigación a nivel universitario, donde nuestro radio de acción es más amplio y donde consideramos es importante hacerlo. Al otro tipo de vinculación al que nos referimos es el asociado a los esquemas clásicos de vinculación académica. A manera de ilustrar los conceptos fundamentales de la interacción de estas áreas presentaremos de manera breve algunos resultados

*Laboratorio de Investigación en Instrumentación, Control, Visión y Robótica. E-mail: llicovir@yahoo.com. Además de los autores los integrantes más activos del grupo son Minerva Rojas, Ricardo Pérez, Miguel Sandoval, Jorge Ríos, Eduardo Fernández, Ian Etnyre, Gregorio Cachón y Sergio Patrón

sobre 3 problemas fundamentales como son la detección de colisiones, la inicialización de pose y el uso de una interfaz háptica. Así mismo se mostrará como estos tres problemas y sus variantes aparecen dentro del contexto de las distintas aplicaciones científicas en neurofisiología, arqueología, bioinformática. Es importante aclarar que todo lo desarrollado a nivel computacional está hecho sobre el sistema operativo Linux, detalle relevante para el ámbito no solo del uso y acceso a la tecnología computacional sino al desarrollo de la misma. El hecho de utilizar software abierto no va aunado tanto al aspecto económico sino a las reglas del juego en las que está inmerso dicho desarrollo.

2. Desarrollo de herramientas computacionales

2.1. La detección de colisiones¹

En los últimos años, la investigación en robótica ha estado interesada en la manipulación por parte del usuario de objetos virtuales, es decir, de objetos en la computadora que son representaciones de objetos que existen en la realidad. Han sido creados mundos virtuales donde el usuario puede agarrar, manipular o tocar objetos que en la realidad sería demasiado peligroso o de plano imposible². Estos sistemas involucran la fusión de datos visuales con táctiles. Los sistemas que han sido creados con ese fin son denominados sistemas hápticos. En los sistemas hápticos se pretende transmitir al operador sensaciones mecánicas a través de un dispositivo ergonómico cuyo movimiento se calcula con base en la interacción generada por el operador mismo y el mundo virtual. Para la visión humana es fácil calcular la distancia entre objetos visibles; no así, para un sistema de cómputo donde la cantidad de información puede ser bastante elevada. El sistema debe ser capaz no sólo de permitir la visualización de fenómenos de colisión sino también de realismo, de forma tal que lo que el operador sienta corresponda a lo que vea. Por esta razón, el cómputo de distancia entre los objetos que participan en el mundo virtual y que son guiados por el operador, es fundamental para el buen desempeño de la interfaz háptica (más adelante dedicamos una sección entera a dichos sistemas). Una vez calculadas las distancias y penetraciones eventuales entre los objetos, es necesario transformar dichas métricas en fuerzas que serán, al instante, aplicadas a la interfaz háptica. Dichas fuerzas requieren ser controladas con un retraso bastante corto; de otra forma, el operador podrá fácilmente tener sensaciones falsas e incoherentes de lo que visualiza. Particularmente, nos interesa encontrar un algoritmo de detección de colisión que nos permita además de detectar la colisión de un objeto respecto a otro, pueda detectar la colisión de un objeto consigo mismo, es decir, entre las partes que lo componen. Tomemos como ejemplo, la representación de un cuadrúpedo que camina: Tanto deseamos que el algoritmo detecte la colisión entre el cuadrúpedo y los obstáculos existentes en el mundo virtual, como entre las piernas del mismo. Por otro lado, sería interesante trabajar con objetos deformables, objetos que resintieran la fuerza aplicada por el usuario y cambiaran su apariencia (se encogieran, por ejemplo) como reacción a ella.

¹Este trabajo fue realizado principalmente por Eduardo Fernández dentro de sus trabajos de tesis de licenciatura. Eduardo es actualmente estudiante de la maestría en Ciencias Matemáticas de la Facultad de Matemáticas de la UADY

²Para más referencia revisar la página del dispositivo PHANTOM, www.sensable.com

2.1.1. Estado del arte en la detección de colisiones

Uno de los problemas fundamentales en la manipulación de objetos dentro de un mundo virtual es, disponiendo de la localización, posición y orientación de cada uno de los objetos que interactúan en él, poder mantener continuamente las distancias entre ellos para saber por dónde pueden moverse los objetos respetando las restricciones de colisión y de espacio existentes en el mundo físico. Podríamos definir colisión como el instante en el que dos objetos entran en contacto físico. En general, si tenemos un par de objetos pueden presentarse tres casos que indiquen la posición relativa de uno respecto al otro: que estén separados, que estén en colisión o contacto, o que exista un grado de penetración entre ellos [1]. Dependiendo del modelado matemático que se utilice para representar los objetos en cuestión, se requiere una estrategia de cómputo para llevar a cabo la medición de las distancias, en principio, asumiendo las posiciones y orientaciones de los objetos como estáticas y posteriormente, nos interesará calcular dicha distancia, teniendo a éstos en movimiento. De igual forma los datos de salida que proporcionan los algoritmos pudieran ser simplemente el cálculo de la distancia entre los objetos, o bien, las coordenadas de los puntos más cercanos del par de objetos, y la profundidad de penetración si fuera el caso. Todos los aspectos mencionados en este apartado son necesarios considerar cuando se evalúan las características de los algoritmos de Colisión, para seleccionar aquél que mejor se ajuste a nuestras necesidades.

2.1.2. Representación de los objetos

Para representar un objeto real en la computadora se emplean modelos matemáticos, entre los cuales encontramos: B-rep, *Boundary Representation* (o Representación de Límites) y CSG, *Constructive Solid Geometry* (o Geometría Sólida Constructiva). Con B-rep describimos al objeto mediante sus caras, aristas y vértices, y un grafo de adyacencias. Con el CSG lo describimos usando un conjunto de primitivas (esfera, cono, cilindro, por ejemplo). Por cada primitiva se especifican sus dimensiones, su marco de coordenadas local y su posición dentro del marco global. Se requiere balancear la cantidad de recursos empleados para modelar los objetos con la exactitud de los datos de salida obtenidos [2]. Existe otro esquema denominado Representación con partición del espacio, en el que un sólido se divide en varios sólidos que no tienen intersección entre sí [3]. Una distinción geométrica importante en la representación de objetos es si el objeto es cóncavo o convexo. Un objeto es convexo si dos puntos cualesquiera dentro del mismo, están unidos por una línea situada por completo dentro del objeto. Un objeto cóncavo no cumple dicha condición. El inicio del estudio de la detección de colisión se refirió sólo al manejo de objetos convexos [1, 2, 4, 5, 6]. Sin embargo, una buena cantidad de objetos en el mundo real emplean modelos cóncavos. ¿Cómo representarlos cuando necesitamos calcular la distancia entre ellos? Entre los más importantes enfoques tenemos: convex hull, o encontrar el conjunto convexo más pequeño que contiene a todos los puntos del objeto; objetos envolventes, o emplear una primitiva que mejor se ajuste al objeto deseado. Una colisión es encontrada, cuando el convex hull o los objetos envolventes se encuentren encimados; sin embargo, los objetos originales no están encimados. Por tanto, el par de puntos más cercanos encontrado es una aproximación de la distancia real [7] y no es la verdadera. Al usar el esquema de representación CSG, y cubrir el objeto deseado con tantas primitivas, de modo que se abarque la mayor cantidad de

su espacio real, surgen conceptos como el de esferas dinámicas, que son volúmenes compuestos por infinidad de esferas, cuyos radios y posiciones varían linealmente a través de la extensión del objeto. La fragmentación de un objeto en sus primitivas consume demasiado tiempo y es exhaustiva. En nuestro caso, la descripción de un objeto es a través de una lista de objetos convexos que conforman el objeto real. Para encontrar la colisión entre un par de objetos, se deberá buscar la colisión entre cada objeto convexo de la jerarquía del primer objeto con cada objeto convexo de la jerarquía del segundo.

Naturaleza de los algoritmos Los algoritmos de cálculo de distancias son por lo general algoritmos geométricos cuya principal naturaleza algorítmica consiste en la búsqueda exhaustiva de entre las características que representan al par de objetos, de aquella que contiene la distancia mínima entre ambos [1, 2, 4, 5, 6]. Como ilustración asumamos un caso simple. Supongamos que nuestros objetos están representados por líneas: bastaría pues, determinar si las líneas que componen un objeto se cruzan en algún punto con las líneas del otro. Si la respuesta del número de líneas que se intersectan es nulo, los objetos deben encontrarse a cierta distancia mayor que cero. De esta forma habrá que encontrar qué par de líneas se encuentra a la menor distancia, calculando la distancia entre todas las líneas y guardando a cada paso el par que me ofrece la menor distancia. Todos los algoritmos que emplean el enfoque anterior son denominados algoritmos de detección de colisión basados en objetos (*Object-based collision detection, OBCD*). Existe otro enfoque, los algoritmos basados en imágenes (*image-based collision detection, IBCD*) que ya no se basan en buscar el par de características cercanas de un par de objetos, sino que se basan en proyectar la geometría del objeto en la imagen de un plano y realizan la prueba de colisión en dicho espacio dimensionalmente reducido [8].

Unidad de tiempo En un ambiente virtual, generalmente los objetos se encuentran en movimiento y se calcula la colisión entre dos unidades de tiempo consecutivos. Los algoritmos más efectivos son aquellos que emplean coherencia de frame, es decir, la diferencia de distancia a la que se mueven los objetos entre dos unidades de tiempo consecutivas es pequeña. Para este caso, el algoritmo toma como datos de entrada los datos de salida proporcionados en el paso previo. Sin embargo, si los objetos se encuentran lejanos se vuelve innecesario evaluar el algoritmo de colisión a cada paso, y surge el concepto de paso de tiempo adaptativo, es decir, una unidad de tiempo que será mayor cuando la distancia entre los objetos esté por encima de un umbral, y menor y constante cuando se encuentre por debajo de ella.

Determinación del par de objetos a evaluar En un mundo virtual donde coexisten tal vez infinidad de objetos, existe una cantidad igual a la combinación de los n objetos tomados de dos en dos de pares a los que se tendría que determinar si existe colisión entre ellos. Resultaría demasiado caro computacionalmente evaluar todos los pares de objetos. Un enfoque más inteligente sería emplear un algoritmo para determinar cuáles son los objetos más cercanos a un objeto dado A y por tanto más probables de estar en colisión con él, y encontrar la distancia sólo entre dichos pares. Así surgen las dos fases, presentes en la mayoría de los algoritmos de Detección de Colisión:

1. **Broad phase.** Se identifican de entre todos los pares del mundo virtual, aquellos que pudieran estar en colisión
2. **Narrow phase.** Dado el par de objetos que pudieran colisionar, determinar si ésta realmente ocurre.

Para la *broad phase* podríamos proceder usando el Algoritmo de Mirtich o de cajas límite alineadas a los ejes (*axes-*

aligned bounding boxes AABB): Envolver cada objeto en alguna primitiva. Teniendo los datos de las coordenadas de las primitivas que representan a cada objeto, se procede a encontrar las intersecciones entre ellas, respecto a cada plano. Si las proyecciones del objeto A respecto al objeto B correspondientes a cada uno de los ejes se sobrepasan entre sí, dicho par potencialmente puede estar en colisión. Por tanto, dicho par se agrega al conjunto de pares de objetos a evaluar su distancia de separación, el cual es mucho menor que el conjunto total de pares del mundo virtual. Para la *narrow phase* se aplicaría alguno de los algoritmos estudiados, mencionados posteriormente.

Datos de salida El dato de salida más simple que puede devolver un algoritmo de detección de colisión es la distancia euclidiana d entre el par de objetos cuando éstos se encuentran separados, la cual siempre es positiva mayor que cero [2] [4]. Cuando dos objetos se encuentran interpenetrados, nos interesaría conocer qué tanto se requiere mover un objeto del otro para que estén en una posición de contacto. Esa distancia es llamada distancia de penetración (*penetration depth*). Existen algoritmos que proporcionan una aproximación de dicha distancia dada por un máximo y mínimo posible [5] [1], y otros que devuelven la distancia exacta [7]. Estos últimos devuelven una distancia positiva que representa la distancia cuando dos objetos se encuentran separados y negativa cuando existe interpenetración. Además, es posible obtener las coordenadas de los puntos en los objetos A y B que delimitan dicha distancia d . Cuando los algoritmos emplean objetos representados usando *B-rrp* es posible obtener el par de características más cercanas entre ellos (la cara de un objeto y la arista del otro, por ejemplo). Para manipular objetos en una interfaz háptica, se requiere conocer la distancia exacta entre dos objetos, y no una mera aproximación, para que la simulación sea la más real posible. En el caso en que se manejen objetos deformables, requerimos conocer la distancia de penetración (o qué tanta resistencia puede un objeto ejercer contra una presión) y con base en ella, realizar las deformaciones del objeto necesarias.

2.1.3. Metodología

Para encontrar el algoritmo que nos ofrezca la funcionalidad deseada, iniciamos con la *narrow-phase* de la detección de colisión. Los algoritmos estudiados se presentan por su aparición cronológica. Uno de los primeros algoritmos de detección de colisión en ser reconocido fue el desarrollado por Lin y Canny [2, 4] en 1991. Es un método iterativo descendente para encontrar los puntos más cercanos y la distancia de separación entre un par de poliedros convexos, basándose en encontrar en cada iteración un par de características más cercano entre ambos poliedros, que el par de características encontrado en la previa iteración. Se basa en el concepto de *boundary* y *coboundary* de cada característica y en la región de Voronoi [2, 4]. Su complejidad es constante dado un procesamiento previo de la representación de los objetos, en cuanto al número de vértices y caras. Su función principal es evitar la colisión. Cuando los objetos se encuentran muy cercanos uno del otro, su desempeño disminuye notablemente. Sin embargo, el algoritmo es muy ilustrativo: enumera todos los casos posibles de contacto entre las características pertenecientes a cada objeto (criterios de aplicabilidad: punto-vértice, punto-arista y punto-cara) (Fig. 1). Es exhaustivo; sin embargo, da una muy buena primera visión y entendimiento del tema de colisión. Al no satisfacer nuestros requerimientos, investigamos el algoritmo GJK. El algoritmo de Gilbert, Jhonson & Keerthi [1, 5] (GJK, por las siglas de sus autores) es un algoritmo para encontrar la distancia

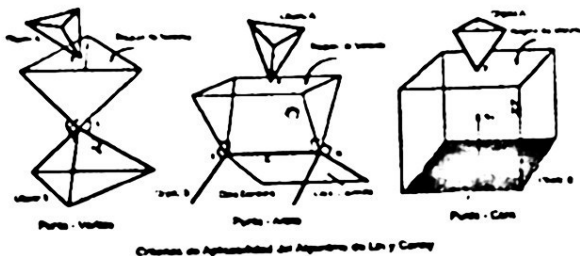


Figura 1: Criterios de aplicabilidad del algoritmo de Lin y Canny

de separación real entre un par de poliedros convexos. Para tal efecto, encuentra el punto en el TCSO más cercano al origen, construyendo en cada iteración el simplex contenido en el TCSO más cercano al origen, que el simplex contruido en la previa iteración. Se basa en los conceptos de suma de Minkowski, espacio de configuración, TCSO (Espacio ob- stáculo de configuración traslacional), punto de apoyo y sim- plex. El algoritmo termina tan pronto como el estimado del error en la distancia calculada sea menor que una tolerancia dada. El atractivo del algoritmo GJK yace en su simplicidad, que lo hace fácil de implementar. Puede manejar eficiente- mente el caso de contacto. En el caso en que los objetos se interpenetran devuelve un simplex a partir del cual se puede calcular un estimado de la distancia de penetración tomando un límite superior e inferior. Sin embargo, tiene una complejidad cuadrática, dada por el número de vértices de cada objeto del par a evaluar. Si consideramos la cantidad de operaciones a realizar en un mundo con una gran cantidad de objetos, el desempeño del algoritmo decrece demasiado. La variación realizada por Cameron al GJK [1] permite, da- do un punto en un simplex, obtener el siguiente punto de apoyo empleando un método de búsqueda *hill-climbing*, re- duciendo así el número de puntos a evaluar. De este modo, la complejidad del algoritmo se vuelve lineal. Esta compleji- dad junto con la simplicidad del GJK, hacen este algoritmo muy atractivo para usar. Debido a que para los objetos de- formables necesitamos conocer la medida de penetración, se siguió investigando y se encontró la variación realizada al GJK por Gino Van Bergen [7] denominada ISA-GJK o al- goritmo *expanding polytope*, que no devuelve un estimado sino la distancia real de penetración, empleando una sub- división de las caras generadas por el simplex que retorna como salida el algoritmo GJK original. Los algoritmos de Lin y Canny, y la variación al GJK realizada por Cameron han sido implementados usando el lenguaje C y la plataforma de visualización Geomview para Linux. Actualmente esta- mos trabajando con la variación al GJK realizada por Van Bergen. Evaluando la complejidad y el desempeño Programación de la Interfaz Es posible cargar en el programa un objeto cuya representación use los formatos OFF, MESH, LIST o INST del Geomview. Por cada objeto se almacenan las coordenadas de sus vértices, aristas y caras. Así como las estructuras necesarias para ejecutar los algoritmos: un grafo de adyacencias y un simplex en el caso del GJK [1, 5], y el par de características más cercanas para el de Lin y Canny [2, 4]. Para detectar la colisión, se requiere de una estruc- tura para cargar todos los objetos del mundo virtual con sus transformaciones actuales y acumuladas, de modo que en un momento dado le pueda indicar el par de objetos a detectar la colisión, sin importar si alguno de los objetos fuera un objeto simple, compuesto o si se trata de una parte, y pueda aplicar el algoritmo respectivo. La estructura empleada fue

un árbol, donde el nodo raíz representa al mundo virtual y cada nodo hijo representa los objetos de dicho mundo. Si el objeto A se compone de otros objetos, el nodo que repre- senta a A se convierte en padre de cada uno de los nodos que representan sus partes, y así sucesivamente. En otra es- tructura, se almacena por cada par de objetos a calcular la colisión, la distancia entre ellos, la cual es representada vi- sualmente usando un segmento amarillo que une los puntos más cercanos entre el par de objetos. Se implementaron dos funcionalidades: permitir a un objeto seguir a otro, es decir, determinar su transformación para que la distancia entre el- los disminuya; y la respuesta de un objeto ante una colisión. Si un objeto colisiona a otro cuyo peso es menor a una con- stante, el otro objeto podrá ser movido en la dirección de contacto, y el primero experimentará un efecto de rebote; de otro modo, el segundo objeto no se moverá por más que se mueva al primero.

2.1.4. Aplicaciones de los Algoritmos

Algunas de las aplicaciones a mencionar realizadas con los algoritmos de Detección de Colisión son:

Simulación de un ambiente virtual. Se imple- mentó un mundo virtual sencillo, en el que una pinza que representa un robot pueda agarrar un objeto y moverlo, re- spectando las restricciones físicas. La pinza fue modelada us- ando dos objetos convexos, una por cada extremo. De modo que el algoritmo busca siempre la distancia de ambos obje- tos con todos aquellos que forman parte de las definiciones de pares realizadas previamente. Además puede soltarlo y el objeto colisionará con el suelo (Fig. 2).

Seguimiento de un objeto. Se cargan dos objetos, y se generan las transformaciones necesarias para que uno de el- los pueda seguir el movimiento del otro, hasta alcanzarlo (Fig. 3).

Interfaz háptica. Se está trabajando con un joystick que tiene sensor de fuerza para que reconozca cuando el objeto que lo representa en el mundo virtual ha colisionado con al- guno del mundo virtual, y responda con la funcionalidad de COLISION explicada previamente.



Figura 2: Movimiento del robot guiado por el detector de colisiones

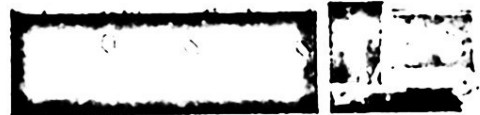


Figura 3: Seguimiento de un objeto por otro y la interfaz háptica

En cuanto a algorítmica, se podrían emplear los enfoque *convex hull* o el de objetos envolventes para representar los objetos, así como un algoritmo que emplee paso de tiempo adaptativo. La detección de colisión es parte esencial en el desarrollo de una interfaz háptica. Se pretenden integrar los algoritmos con una interfaz ergonómica mediante la cual el usuario pueda experimentar la colisión con un objeto virtu- al, y una fuerza de reacción. Otra aplicación sería emplear

un modelo de un sistema multiarticulado que camine en un mundo virtual respetando la colisión, y que pueda realizar el seguimiento de otro objeto empleando los datos de salida de los algoritmos presentados. El estudio de la detección de colisión es un tema que va a estar en boga por mucho tiempo. Poder estudiar las diversas maneras de abordar el problema es realmente enriquecedor.

2.2. Simulación y Emulación de la locomoción³

Hoy día podemos atestiguar que uno de los concursos de robótica más populares es sobre el de las máquinas caminantes. Después de arduas horas de trabajo, vemos algunos proyectos de ingeniería cuya mayor gracia consiste en el haber diseñado armatostes mecánicos que son capaces de moverse de un punto a otro a través del movimiento de sus patas, cuya fuerza podría cargar sin lugar a dudas a más de una persona. Pero que en el menor conflicto con la gravedad ese tipo de sistemas no tienen la menor alternativa para equilibrarse. Nuestro objetivo es trabajar en sistemas al borde de la inestabilidad y lograr, computacionalmente, generar que camine y así mismo su estabilización. El objetivo final es hacer una herramienta a la cual se le puedan introducir corrientes eléctricas y podamos observar el movimiento de forma interactiva tanto en simulación como físicamente.



Figura 4: Sistema de simulación y el sistema físico

La simulación de sistemas multi-articulados es complicada pues demanda el expresar de forma explícita la coordinación de las partes incluyendo la solución a las restricciones cinemáticas impuestas por el sistema. En particular nos interesa desarrollar plataformas de desarrollo que sean abiertas y flexibles sobre las cuales se puedan implementar distintos algoritmos tanto de planificación de movimientos como de integración numérica. Se ilustran los avances del proyecto con la emulación de un cuadrúpedo cuyo comportamiento dinámico se aproxima, en un primer término con sistemas cuasi-estáticos. La plataforma propuesta conforma la base sobre la cual futuras investigaciones pueden llevarse a cabo y contemplar aspectos de calibración y sistemas dinámicos. Desde los principios de la informática hubo un interés claro en la utilización de sistemas computacionales para simular y emular sistemas complejos. Simular (del lat. *simulare*) refiere a la representación de un cosa mientras que emular (del lat. *aemulare*) hace referencia a imitar las acciones de otra entidad procurando igualarlas e incluso excederlas. En el presente trabajo nos interesa explotar al máximo las propiedades del sistema desarrollado en ambas direcciones. Cheriton refiere a una nueva generación de programación. En esta nueva generación, Cheriton refiere a que

las nuevas aplicaciones deberán ser estructuradas como simulaciones de sistemas y la programación será enormemente concerniente con el desarrollo de modelos. La aparición de sistemas emuladores de comportamientos cinemáticos de robots tuvo su auge en los 80's donde fue posible desarrollar las primeras aplicaciones de robótica industrial on line. Inclusive a nivel académico empezaron a aparecer los primeros integradores numéricos sobre el modelado de sistemas dinámico. En la posterior aparición de plataformas en tiempo real se comenzó a requerir de poder de modelado para integrar tanto aspectos de simulación como de solución numérica. Hoy día existen algunos paquetes comerciales que permiten el modelado de sistemas robotizados complejos, sin embargo no existe aún ninguna plataforma de simulación y emulación de sistemas dinámicos donde se pueda interactuar con dispositivos conectados al sistema computacional que se encarga de llevar a cabo la simulación. En la actualidad han aparecido necesidades intrínsecas a la interacción hombre-máquina donde cada uno de los componentes del sistema deben poder ser accesibles de forma transparente. Tal es el caso de los sistemas de realidad virtual donde el éxito principal de las aplicaciones dependen de la buena integración de las plataformas de visualización e interacción electromecánicas. Generar los patrones de movimiento (*gaits*) de cada una de las patas a partir de la definición de las trayectorias de la punta del pie o de algún punto en particular, requiere la definición de la cinemática inversa. En contraparte, la cinemática directa mapea las configuraciones articulares al correspondiente espacio tridimensional. Para generar dicho mapeo es necesario asignar una serie de referenciales coordinados a cada uno de los componentes del sistema. Un método comúnmente usado en robótica es el desarrollado por Denavit y Hartenberg donde a partir de transformaciones homogéneas y un número mínimo de variables es posible definir el comportamiento cinemático de los referenciales que representan al sistema. Cada articulación es numerada de forma ascendente partiendo de la base (en nuestro caso de la base de la cadera) hasta la parte más lejana. Por simplicidad consideraremos el cuerpo del cuadrúpedo como el punto común donde convergerán las cuatro cadenas cinemáticas. Las cadenas cinemáticas están compuestas de 2 eslabones y 3 grados de libertad en cada extremidad del robot. Después que se hubo determinado tal configuración, se escribieron funciones que permitieran calcular las matrices de transformación adecuadas y que pudieran escribirse a la salida estándar en el lenguaje que interpreta Geomview. Obsérvese en la fig. 2 el esquema del sistema multi-articulado que representa a las patas y en la tabla 1 los parámetros para la configuración de la pata del robot cuadrúpedo, dicha tabla es la misma para las cuatro patas. Para el robot Aibo, existe un deslizamiento que se representaría con el parámetro d de la segunda articulación pero por simplicidad para los cálculos se consideró con valor 0. Los parámetros i y ai que corresponden a la configuración espacial del sistema son fácilmente modificables a partir de un archivo de configuración que el sistema lee cuando inicia su ejecución. Los parámetros i corresponden a las variables articulares. En total obtuvimos seis sistemas articulados que se mantienen unidos a la base del robot, cuatro para patas, una para la cola y una más para la cabeza. En la cabeza también se tienen 3 grados de libertad, en la cola solamente uno, a pesar que estos dos últimos sistemas no son cruciales para el desarrollo de caminatas si era importante tomarlas en cuenta ya que se partió del modelo del robot Aibo5. Una vez que el sistema ya tenía la cinemática directa funcionando correctamente, el siguiente paso era agregarle la cinemática inversa, hay muchas formas de hacerlo, escogimos el método

³ Este trabajo fue realizado principalmente por Jorge Ríos dentro de sus trabajos de tesis de licenciatura. Jorge es actualmente estudiante de la maestría en Ciencias Matemáticas de la Facultad de Matemáticas de la UADY. El diseño y construcción del cuadrúpedo ha sido desarrollado por Ricardo Pérez, becario Promep, quién estudia el noveno semestre de la Licenciatura en Ciencias Computacionales dentro de la Facultad de Matemáticas

geométrico dado que sólo teníamos tres grados de libertad en las patas, y no era tan complicado obtenerla.

2.2.1. Desarrollo de la plataforma

El ciclo de desarrollo de software bajo la perspectiva actual comprende la inserción dinámica sobre los requerimientos que expanden el diseño después de su conclusión: lo que nos da una nueva implementación que será probada, y generará un nuevo producto, el cual al ser analizado produce nuevos requerimientos. Nuestro objetivo final es la completa emulación de sistema multi-articulados sobre los cuales se puedan llevar a cabo experiencias de diseño, análisis cinemático, dinámico, control, planificación de movimientos, interfaces hápticas etc. Sólo sobre un modelo computacional completo se puede tener expectativas claras en el desarrollo y reutilización del software. Hornby menciona la necesidad de una plataforma de experimentación computacional donde se puedan realizar experimentos. En particular, la experiencia descrita menciona la utilización del robot Aibo para desarrollar estrategias de caminata en robots cuadrúpedos. Se mencionaba que no contaban con un simulador para realizar sus experimentos y que trabajaron directamente con el robot físico. Como plataforma base de este trabajo tenemos al programa Geomview el cual sirve como despliegue de nuestros cálculos y experimentos. Se usa este programa porque nos evita el tener que preocuparnos por las gráficas y el manejo de los objetos en 3D. Es un programa bastante probado y adecuado para la simulación, muy flexible y con características de manejo de cámaras que lo hacen un visor 3D muy completo.

2.2.2. Detalles sobre el modelado

Desde el principio se planteó la necesidad de contar con una herramienta para dibujar objetos que sirvieran como piezas de nuestro sistema multi-articulado y que pudieran visualizarse en geomview. Geomview tiene la característica de comunicarse con módulos externos escritos en cualquier lenguaje de programación, el único requisito es que el módulo externo envíe comandos a la salida estándar y que dichos comandos estén en un lenguaje parecido al LISP llamado gcl (Geomview command language). Si Geomview puede interpretar los comandos que lee de la salida estándar, ejecuta las órdenes que el módulo haya pedido. La implementación del software se realizó programando en lenguaje C y C++ mediante Kylix 3.0. Se eligió este programa porque ofrecía un ambiente de programación y depuración sencillo y funcional, para el programa de simulación se ofrece además una versión reducida que requiere solamente un compilador de C y C++. El software modelador no se comunica directamente con Geomview, su función es tomar parámetros que el usuario proporciona relacionados con la altura, la longitud, etc. de las piezas y crea los archivos necesarios para que se puedan visualizar posteriormente los objetos definidos. Se usaron las rutinas con las que ya se contaba y se crearon algunas más para dibujar objetos complejos. A partir de las especificaciones de las piezas del robot Aibo se diseñaron las piezas del sistema multiarticulado, por supuesto bastante simplificadas ya que nuestro interés principal no era reconstruir el Aibo detallado sino tener un sistema cuadrúpedo que cumpliera reglas cinemáticas. Se implementó una técnica sencilla para dibujar sólidos la cual consiste en dividir la figura que se quiere modelar en tres partes, una cara superior, una cara inferior y una parte que uniera los bordes de estas dos caras completando así el objeto sólido.



Figura 5: El sistema se puede ir armando paso a paso

2.2.3. Comunicación

Para representar ambas caras usamos los formatos off6 y para representar a la parte envolvente usamos el formato mesh. Para darle un poco más de variedad a los objetos que queríamos representar, hubo que programar una función para dibujar líneas curvas basado en las funciones de Bezier, la rutina que se programó resultó bastante útil, no sólo para dibujar piezas sino también para definir trayectorias curvas de los miembros del sistema multi-articulado. Geomview permite agrupar objetos y organizarlos de manera jerárquica, esta característica fue útil para representar las piezas, una vez que teníamos creado los archivos off y mesh para una figura había que agruparlas bajo un mismo nombre, el formato list nos permitió esto. Como se quería aplicar transformaciones sobre los sólidos se empleó el formato inst. Ambos formatos son adecuadamente generados por el programa modelador. Con el programa modelador podemos dibujar todas las piezas de nuestro sistema multi-articulado y especificar el tamaño de éstas, pero no se limita a eso sino que se podrían combinar las rutinas y crear nuevos objetos. El programa simulador es el que se comunica directamente con geomview, le da instrucciones para que cargue los objetos y los despliegue, permite especificar una configuración inicial y tiene botones para hacer que el cuadrúpedo ejecute una caminata con una longitud también especificada por el usuario. Cuenta con una vista que nos permite saber la posición de cada uno de las cuatro patas y del centro del cuerpo. Además se puede introducir comandos que ejecutan las mismas acciones que los botones, es un conjunto de instrucciones bastante reducido pero se espera en un futuro se mejoren y se usen para probar algoritmos de caminata. Cuando el programa modelador genera los puntos para los sólidos que se modelan lo hace tomando en cuenta el sistema de coordenadas de Geomview, cuando nosotros definimos la cinemática suponemos que cada pieza se encuentra definida con respecto a su propio eje de coordenadas con las orientaciones dadas por los parámetros DH, lo cual es incorrecto, por eso antes de correr el programa hay que ajustar las piezas con una utilidad que se desarrolló. Para el programa simulador se definió una clase llamada SistArti, que representa a cada cadena cinemática que forma el modelo del cuadrúpedo. Se definió también una clase RoboCuad que representa a un conjunto de cadenas cinemáticas que construyen un robot cuadrúpedo. La posición del robot y la de cada elemento del sistema se define con base a la posición del centro del cuerpo. A la hora de definir la forma en que se iba a coordinar los movimientos necesarios para que el sistema se desplace hubo que tomar en cuenta que, aunque en este momento no se implementaron propiedades dinámicas, sí debía mostrar estabilidad, lo que se hizo fue plantear un algoritmo de caminata para el cual siempre se mantuviera tres patas en el piso, además en la configuración inicial con que se hicieron las pruebas se trató de que las piernas estén flexionadas desde el principio como efecto del peso del robot. En la simulación se cuenta con un parámetro que representa

la longitud del paso que cada pata puede dar, asimismo hay un parámetro que indica que tanto puede levantar esa pata, esos valores se emplean para definir los puntos de control de la curva de bezier que genera la trayectoria del efector final. El efector final no se encuentra en la punta de la pata sino en el tobillo. Se debe ajustar el número de puntos que se generan en la trayectoria para dar un paso a fin de dar una apariencia más natural a la caminata, actualmente genera 10 puntos.

2.3. Retroalimentación Visual por Computadora⁴

Usar cámaras de video como sensores comienza a ser una realidad debido al enorme avance tecnológico en la electrónica de consumo. Sin embargo, toda la ola de especulaciones en torno a la inteligencia artificial han arrojado más que falsas expectativas sobre lo que podemos esperar de un sistema "inteligente". En la figura siguiente se muestra un problema real de automatización de un proceso de extracción de piezas de metal de un horno industrial. Se pretende integrar al sistema de control de la pinza que se usa para sacar las piezas del horno un sistema de visión que permita prescindir de la ayuda de un operador para realizar dicha tarea. En el sistema de visión se contemplan problemas como la inicialización de pose y seguimiento del objeto. Se muestra cómo se pretende abstraer la información necesaria para guiar a la pinza hacia su objetivo a partir de imágenes de una cámara de video. Después de ya varios años de la irrupción de sistemas de almacenamiento y procesamiento masivo de información tal parece que nuestras licuadoras de datos no son tan aptas para procesar cualquier tipo de información para transformar cadenas de datos en valores significativos para situaciones reales[9]. Considerar una cámara de video como un sensor autónomo, tal como ahora un potenciómetro nos indica variaciones de resistencia en un carga, es literalmente imposible. Poder interactuar con una cámara conectada a una computadora e interrogarla sobre el tipo de objetos que observa y no digamos, que nos mida, el espacio que observamos, es una de las ambiciones más intrépidas de la actualidad. Podemos, eso sí, calibrar la cámara y a través del análisis de señales procesar información proveniente de ella casi en tiempo real.

Calibración Un sistema de visión está compuesto por una o más cámaras de video con la circuitería y programación necesaria para poder adquirir y desplegar la imagen en un monitor, así como extraer la información más relevante de la imagen. El sistema de visión proporciona una o más imágenes bidimensionales del mundo real. La relación entre las coordenadas asociadas en cada imagen bidimensional y las correspondientes coordenadas en el mundo real es descrita en términos de transformaciones de perspectiva y de movimiento rígido las cuales emplean un conjunto de parámetros denominados parámetros *extrínsecos* e *intrínsecos*. Los parámetros *extrínsecos* determinan la posición y orientación de la cámara con respecto a un marco de coordenadas, los parámetros *intrínsecos* determinan las características geométricas y ópticas de la misma. Los parámetros *extrínsecos* e *intrínsecos* son independientes entre sí. El

proceso de obtener el valor numérico de los parámetros de la cámara se denomina calibración y tiene como propósito establecer la relación entre un marco de referencia y sus correspondientes coordenadas en la imagen bidimensional vista desde el monitor. Con el proceso de calibración ya tendremos forma de relacionar el sistema de coordenadas de la imagen con el sistema de coordenadas del mundo real.

2.3.1. Inicialización de pose

El problema de inicialización de pose consiste en determinar por medio de cámaras de video la posición de un objeto fijo en un ambiente tridimensional. Este es un problema complejo ya que solo se cuenta con imágenes bidimensionales del mundo en las que el objeto a localizar se encuentra rodeado de todos los demás elementos que conforman el ambiente y que nos pueden crear varios problemas al tratar de encontrar el objeto en la imagen como son: oclusiones (una parte del objeto está oculta), ambigüedades (objetos que se parezcan al que buscamos), distorsiones de la imagen debido a la luz, etc. El proceso de inicialización de pose es fundamental para resolver diferentes problemas de la Visión Computacional, como son el seguimiento de objetos y la manipulación de objetos, ya que para poder realizar alguna de estas actividades, debemos saber exactamente (o tener al menos una aproximación de) dónde se encuentra el objeto que queremos seguir o manipular. El algoritmo que se presenta a continuación resuelve este problema de una forma muy eficiente y confiable.

2.3.2. El algoritmo de Wunsch e Hirzinger

Wunsch e Hirzinger [10] presentan un algoritmo para registrar modelos en 3D de una sola o de una secuencia de imágenes 2D. Como las imágenes de la cámara no producen una información completa de las 3 dimensiones la idea clave propuesta es explotar la perspectiva inversa, formada por los rayos de proyección determinados por el punto focal de la cámara y los puntos característicos en la imagen, para buscar la correspondencia en el espacio 3D en vez de hacerlo en el plano como en otros algoritmos. Esto no sólo simplifica la estimación de los parámetros de movimiento, si no que también reduce las ambigüedades de apareamiento debidas a las oclusiones. Es importante mencionar que se conoce de antemano el modelo CAD del objeto (su tamaño y características). El algoritmo se compone de 2 etapas iterativas, la primera determina la correspondencia entre las características de la imagen y los datos del modelo, mientras que la segunda calcula una transformación de un cuerpo rígido en 3 dimensiones que minimice el desplazamiento de los puntos apareados, es decir, que acerque el modelo virtual del objeto a la posición real del mismo. Los pasos a seguir en este algoritmo son: 1.- Obtener los puntos característicos de la imagen, puntos que nos den información relevante del objeto, por ejemplo las esquinas. 2.- Pasar esa información que



Figura 6: Ilustración del problema de inicialización de pose en un ambiente industrial estructurado

⁴ Este trabajo fue realizado principalmente por Minerva Rojas dentro de sus trabajos de tesis de licenciatura. Minerva es actualmente estudiante de la maestría en Ciencias Matemáticas de la Facultad de Matemáticas de la UADY. Ramón Atoche, profesor del Instituto Tecnológico de Mérida y estudiante de Doctorado conjunto entre el IT-Mérida y la Universidad Politécnica de Valencia, ha sido pieza fundamental en el desarrollo de este trabajo, principalmente en las perspectivas sobre el uso de lo aquí desarrollado con FPGA's

tenemos en el plano bidimensional al espacio tridimensional (crear los rayos de proyección). 3.- Establecer correspondencias entre los puntos del modelo CAD y las líneas proyectadas (CLP)[11, 12]. 4.- Calcular la rotación y traslación necesarias para acercar el modelo CAD a las líneas proyectadas (SVD). 5.- Aplicar la rotación y traslación calculadas al modelo virtual (Registro del modelo). 6.- Calcular el error y los pesos que nos ayudan a controlar las falsas correspondencias de los puntos (M-Estimators). 7.- Regresar al paso 3 hasta que el error sea suficientemente pequeño. Al terminar, el algoritmo nos devuelve la posición del objeto en el espacio tridimensional, ya que pusimos al modelo virtual en una posición conocida y podemos saber exactamente las transformaciones que se le aplicaron, por lo que ahora sabemos en que posición está el objeto virtual (que debe ser muy parecida a la posición del objeto real). En las siguientes secciones del artículo describiremos más detalladamente cada paso del algoritmo.



Figura 7: a) imagen original b) Canny c) transformada de hough para líneas

Puntos característicos Para encontrar las esquinas dentro de la imagen utilizamos un método conocido como la transformada de Hough[13]. Se utiliza para el enlace de puntos de borde y la extracción de rectas. Implica la transformación de coordenadas cartesianas a coordenadas polares. El concepto básico que se utiliza para la detección de líneas en la transformada de Hough es la dualidad punto-línea. Sabemos que la ecuación de una línea recta está dada por: $y = mx + c$. En esta ecuación estamos utilizando el espacio $x - y$, pero esta ecuación podemos replantearla en términos de m y de c de modo que en lugar de utilizar el espacio $x - y$ utilicemos el espacio $m - c$ (el espacio de la pendiente y el desplazamiento de la recta): $c = -mx + y$. Un sólo punto en el espacio $m - c$ corresponde a una recta en el espacio $x - y$, y un punto en el espacio $x - y$ se puede definir con el punto donde cortan las infinitas líneas (para todos los valores de m que pasan a través de él). Si tenemos un conjunto de líneas en el espacio $x - y$ las podemos representar como un conjunto de puntos en el espacio $m - c$ (las diferentes pendientes m y desplazamientos de las rectas c). De igual forma, si tenemos varios puntos colineales en el espacio $x - y$, pueden representarse como un conjunto de líneas en espacio $m - c$ que se intersectan en un punto (porque tienen la misma pendiente). Nosotros caracterizamos a los objetos cuya pose queremos encontrar por sus bordes, con tal fin utilizamos el filtro de Canny[14] para procesar la imagen original y obtener una imagen binaria en donde los bordes del objeto estén representados. Es a ésta última imagen a quien aplicamos la transformada de Hough para extraer las representaciones paramétricas de los bordes del objeto. El conjunto de intersecciones entre los bordes encontrados formarán el conjunto de puntos característicos del objeto. En la figura 7 podemos observar el proceso completo de segmentación de la imagen, desde la imagen original, la imagen devuelta por el filtro Canny y la imagen donde se dibujan sobre la imagen original las líneas calculadas por la transformada de Hough.

Rayos de Proyección Los rayos de proyección (paso 2) $P^{-1}(p)$ están determinados por el punto focal de la cámara y P (puntos característicos de la imagen) y se proyectan al infinito como se muestra en la figura 3. Podemos observar que el objeto está "cubierto" por los rayos de proyección,

por lo que podemos poner al objeto virtual en un principio entre los rayos (trasladarlo sobre el eje Z un valor arbitrario) para reducir el número de iteraciones.

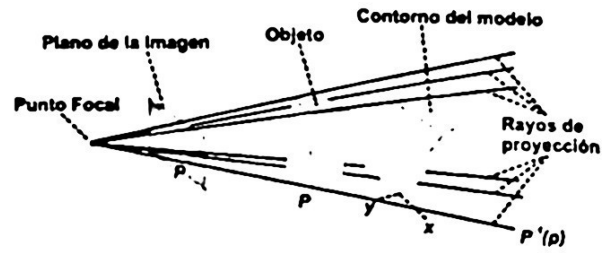


Figura 8: El principio de operación del algoritmo de Wunsch

La ecuación de $P^{-1}(p)$ con respecto al sistema de coordenadas de la cámara localizado en el punto focal está dado por:

$$P^{-1}(p) = \{y/y = \lambda(p_x, p_y, f)^T, \lambda \in \mathbb{R}\}$$

Donde λ es una constante real, f es la distancia focal de la cámara, p_x y p_y son las coordenadas en la imagen de p , escalonados a las mismas unidades de los datos del modelo. Asumiendo que p es la proyección de un punto P en el espacio 3D, entonces P cae en el rayo de proyección, pero la distancia de P al plano de la imagen no puede ser recuperada, por lo que se necesita de los demás pasos del algoritmo.

El Algoritmo del Punto más Cercano (CLP Closest Point Algorithm) Ahora que tenemos el modelo y los rayos de proyección, es necesario que determinemos las correspondencias entre ambos. Para hacer esto utilizamos el CLP. El CLP[15] es un método popular de registro de un modelo a un conjunto de datos 3D y está basado en el algoritmo iterativo del punto más cercano (ICP Iterative Closest Point). Este algoritmo es un procedimiento iterativo donde cada iteración consiste de dos pasos. En el primero, los puntos vecinos más cercanos son puestos en correspondencia, mientras se conserva la pose actual del objeto. El segundo paso actualiza el registro actual minimizando mediante mínimos cuadrados el desplazamiento de los pares de puntos mapeados. El error posicional decrece en cada paso, y el procedimiento converge monótonicamente a un mínimo local de error posicional. Lo que queremos es aproximar correspondencias por asociación de los puntos finales de los segmentos de líneas que representan la más pequeña distancia entre $P^{-1}(p)$ y el contorno de la imagen. Mas formalmente, dado un punto p en la imagen y el contorno del modelo X , el operador:

$$[x, p] = CLP(p, X)$$

regresa un par de puntos 3D asociados x y y , tales que $x \in X$ y $y \in P^{-1}(p)$ (Nota: x está en el contorno del modelo, e y está en la línea de proyección de p).

Descomposición en Valores Singulares (SVD Singular Value Decomposition) Una vez que tenemos las correspondencias entre el modelo y los rayos de proyección, necesitamos encontrar una rotación y una traslación que minimicen las distancias entre ambos, es decir, queremos acercar el modelo virtual a los rayos de proyección. Para esto primero calculamos la matriz de correlación cruzada que nos permite determinar la relación entre el modelo y los rayos:

$$Q_{xy} = \sum w_i (y_i - \bar{y})(x_i - \bar{x})$$

Dónde w_i son los pesos que controlan las falsas correspondencias, y_i es el i -ésimo punto en su correspondiente rayo de proyección, $\bar{y}$ es el promedio de los puntos en los rayos, x_i es

el i -ésimo punto del modelo y $\bar{x}$ es el promedio de los puntos en el modelo. A partir de ésta matriz aplicamos el algoritmo de Descomposición en Valores Singulares (SVD por sus siglas en inglés). SVD es un método que resuelve la mayoría de los problemas lineales de mínimos cuadrados. Los métodos de SVD están basados en el siguiente teorema de álgebra lineal: Cualquier matriz A de $M \times N$ cuyo número de filas M es mayor o igual que su número de columnas N , puede ser escrita como el producto de una matriz de $M \times N$ columnas ortogonales U , una matriz diagonal W de $N \times N$ con elementos positivos o cero (los valores singulares), y la transpuesta de una matriz ortogonal V de $N \times N$. Es decir, podemos descomponer cualquier matriz A en tres matrices U , W y V de forma que: $A = UWV^T$. Las matrices U y V son ortogonales en el sentido de que sus columnas son ortonormales. La descomposición también puede ser llevada a cabo cuando $M < N$. En este caso los valores singulares w_i para $j = M + 1, \dots, N$ son todos cero, y las correspondientes columnas de U son también cero.

Estimadores robustos (M -Estimators) Los métodos de mínimos cuadrados estándares tratan de minimizar los cuadrados de los errores, lo cual no es muy confiable si existen puntos raros (*ouliers*) dentro de la distribución del error. En nuestro caso tenemos que considerarlos debido a que los puntos característicos de más son producidos durante la detección de bordes en la imagen debido a factores ambientales y al error inherente al proceso (e.g. líneas presentes en la imagen que no pertenecen al objeto, multiple detección de bordes). Los puntos raros causan un efecto tan grande en la minimización que los parámetros que se están estimando se distorsionan. Los estimadores M [16] tratan de reducir el efecto de dichos puntos raros por medio de la función ρ que es una función simétrica, definida positiva con un único mínimo en cero y se elige de forma que sea menos incremental que el cuadrado. Dicha función es como sigue:

$$\rho(x) = \begin{cases} 0,5x^2 & \text{si } \|x\| < a \\ a\|x\| - 0,5a^2 & \text{en otro caso} \end{cases}$$

2.3.3. Resultados y extensiones del método



Figura 9: Evolución y vistas de la inicialización de pose

La implementación del algoritmo se realizó bajo el lenguaje de programación C++ para Linux. El proceso de segmentación de la imagen se realizó utilizando bibliotecas

de funciones para procesamiento de imágenes desarrolladas *ex profeso* en el laboratorio. Para poder analizar gráficamente los resultados obtenidos por el algoritmo, utilizamos el visor de geometría interactiva Geomview desarrollado bajo Open GL, y que nos permite interactuar muy bien con programas desarrollados en el lenguaje de programación C. En la figura 9 podemos observar primero una vista de frente de la imagen que nos proporciona la cámara con la que estamos trabajando y sobre la que ya se aplicaron los procesos de segmentación mencionados anteriormente. En ésta imagen se observan los rayos de proyección que ya han sido calculados y también se observa dónde se encuentran los ejes coordenados X y Y . En la imagen del centro podemos observar cómo el objeto virtual se acerca hacia los rayos de proyección y en la imagen de la izquierda el objeto virtual ya llegó a su posición final que es una buena aproximación de la posición real del objeto que se utilizó. Para ilustrar la rapidez con la que el algoritmo converge a una solución óptima, se analizarán las siguientes gráficas³

En la primera gráfica de la figura podemos observar la disminución del error en cada iteración, vemos que en las primeras 10 iteraciones del algoritmo el error decrece rápidamente, mientras que en las demás iteraciones es casi imperceptible su disminución entre una iteración y otra. Es por eso que se utiliza a ΔE dentro de las condiciones de paro del algoritmo, que representa un mínimo de disminución del error entre dos iteraciones. En la imagen del centro de la misma figura se muestra la forma en la que las componentes de la diagonal de la matriz de rotación se van acercando a 1 (ya que queremos que al no encontrar una mejor rotación para minimizar el error, el objeto no se rote, es decir, queremos que la matriz de rotación llegue a ser la matriz identidad). En el caso del vector de traslación queremos que sea el vector Cero, y en la parte derecha de la figura podemos ver que las gráficas de las componentes del vector de traslación, llegan a cero en menos de 25 iteraciones. En este caso se encontró una posición óptima con menos de 100 iteraciones, pero existen casos en los que se utilizan hasta 250 iteraciones, dependiendo de la posición inicial en la que se coloque al objeto virtual y de los mínimos locales que encuentre el algoritmo. Actualmente nos encontramos estudiando 2 posibles extensiones al método originalmente propuesto por Wunsch. El problema de inicialización de pose es complejo. Aún en ambientes estructurados existen problemas inherentes a la naturaleza de los objetos en cuestión. Nos interesa abordar el problema de *tracking* asumiendo el vertiginoso avance de la computación y el procesamiento de señales que se está haciendo presente con la irrupción de las tecnologías de alta integración, en particular los *PPGA's*[17]. Bajo ese esquema nuestra pretensión es instrumentar el problema de seguimiento con secuencias de inicialización de pose. De esta forma aún si el objeto se pierde, será encontrado a la siguiente interacción.

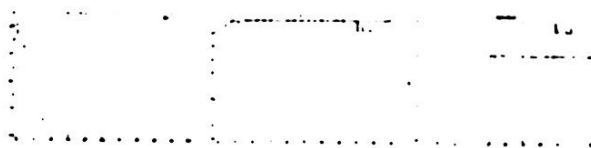


Figura 10: Gráfica del error total, componentes de la diagonal de la matriz de rotación, y componentes del vector de traslación en X , Y e Z respectivamente.

³Aunque el número de iteraciones que realizó el algoritmo fue de 107, solo graficamos 50 iteraciones ya que en las demás observamos un comportamiento similar para cada gráfica a partir de la iteración 50.

3. Aplicación de las herramientas en el contexto científico

3.1. Diseño y construcción de una plataforma computacional para el uso de sistemas hápticos⁶

Tener acceso a las tecnologías modernas parece condicionar el desarrollo científico y tecnológico de los países en desarrollo. El apoyo otorgado por las agencias de financiamiento de fondos para Investigación y Desarrollo se ve mermado cuando la investigación en cuestión tiene la mínima componente tecnológica. La investigación en robótica es una mezcla - algunas veces exótica - de paradigmas concurrentes en ingeniería de control, electrónica, mecánica y computación. Los orígenes de la manipulación diestra se encuentran mezclados con la irrupción de los cada vez más frecuentes intentos por la elaboración de prótesis de manos⁷. Como es lógico, la elaboración de dichos dispositivos fué concebida con la finalidad de proveer al discapacitado la posibilidad de interactuar de manera natural con el ambiente. Las primeras prótesis fueron bastante modestas y subsecuentemente se le fueron agregando características ergonómicas inherentes a las necesidades mismas de las prótesis. Por una época se pretendió integrar al dispositivo propiedades de comunicación muscular (por medio de dispositivos Myoelectrícos); sin embargo dichos dispositivos requerían de un entrenamiento bastante pesado, sin resultados considerables en el desempeño de la manipulación. Dentro del ámbito de la robótica industrial se han considerado órganos efectores relativamente simples - tipo pinza - dentro del contexto del tipo de tareas de prensar y mover (*pick and place*[19]). Sin embargo en los últimos años se han desarrollado órganos efectores *ad-hoc* al tipo de tareas. De esta forma se han separado las necesidades de extensión prácticas de las prótesis por un lado y de las necesidades de la robótica industrial por el otro. Sin embargo, en la actualidad ambos requerimientos tienden a converger hacia el desarrollo de la mano ideal, ya sea por las necesidades de manipulación o por las necesidades más atractivas de interfases hápticas.

3.1.1. De la definición de una tarea de manipulación a su realización

Sobre esta necesidad se ha avanzado de manera considerable con la intención de entender a cabalidad la naturaleza

⁶Este trabajo fue realizado principalmente por Ricardo Pérez y Miguel Sandoval, ambos estudiantes del noveno semestre de LCC

⁷La contraparte científica refiere principalmente al uso de manos articuladas para robótica industrial. Ver Tanie[18] para un extraordinario *overview* en dicha utilización



Figura 11: Principio de manipulación háptica : la mano se mueve al acorde del sistema de interface hombre-máquina e interactúa con una esfera que solo existe en simulación. El comportamiento de dicha interacción es programable.

de la manipulación diestra. Los primeros trabajos serios para el entendimiento de la manipulación diestra provienen de la neurofisiología la cual se mantiene muy activa (ver [20] para un excelente compendio en neurofisiología de la manipulación diestra). Algunas de las aportaciones más importantes y vigentes dentro del contexto de la robótica contemplan desde las aseveraciones acerca de que es el Sistema Nervioso Central el que computa un modelo interno de la dinámica del sistema brazo-mano de tal forma que se pre-calculan las fuerzas y movimientos necesarios para cada tarea, soslayando así la importancia de un sistema de retroalimentación. Desde el punto de vista de control automático esto significaría la necesidad de computar el "regresor" dinámico de nuestros sistema a controlar en tiempo real y a cada interacción. Esto forzaría el definir una estrategia para obtener los parámetros off-line o lo más importante el cómputo de los parámetros en línea, lo que requeriría de un sistema adaptable. La manipulación diestra requiere, por naturaleza, la consideración de propiedades sensoriales bastante finas. Esto involucraría la comprensión de los reflejos supraespinales, área de investigación muy activa dentro de la investigación en neurofisiología. La obvia importancia de la coordinación sensorimotriz en la casi totalidad de las aplicaciones en robótica incluyen la consideración de la retroalimentación visual como un artifice bastante útil tecnológicamente: en la manipulación humana es difícil separar la enorme dependencia de esta. Sin embargo la simplificación presupone una concepción distinta de la fisionomía del sistema de control. La casi mayoría de la investigaciones de los neurofisiólogos interesados en la manipulación diestra han involucrado el diseño de dispositivos interesantes desde el punto de vista ergonómico pues requieren que los experimentos se den de la manera más natural posible. Ahí hay ideas para el diseño de interfases hápticas específicas. Si bien el diseño de dispositivos mecánicos específicos para la manipulación diestra ha generado dispositivos a escala humana y con transmisión de fuerza necesaria para prensar con facilidad una manzana, los problemas operativos inherentes a tal miniturización han castigado de manera importante el avance en la experimentación en dicha área. Sin embargo han irrumpido en escena macro-manos articuladas, producto de una desesperación por lo restrictivo que es trabajar con sistemas pequeños⁸. El desarrollo científico en la manipulación diestra, estrictamente hablando, se han diferenciado en 3 problemas fundamentales: la planificación de movimientos[21], el control de los mismos[22] y la retroalimentación visual de la manipulación[23]. Para cada uno de ellos se pueden matizar la complejidad del modelo a considerar hasta la concepción de un sistema híbrido que permita la incorporación de todos los componentes y definir así la problemática propia de la manipulación diestra[24].

3.1.2. Manipulación háptica

La interacción háptica puntual está empezando a ser obsoleta[25]. Cada vez es más importante nutrir con información de mayor calidad a los sistemas teleoperados. Después de ya muchos años de experimentación, los tele-robotistas

⁸Tan solo 3 prototipos de manos articuladas han logrado colocarse en el mercado, de los cuales uno se vende como tal y no es más que un *gripper* sofisticado (*Barret Hand*); el otro es un dispositivo bastante antropomórfico pero bastante limitado en sus operabilidad (*UTAH-MIT hand*) y la ya bastante popular e ineficiente, *Salisbury Hand*. Las manos más avanzadas tecnológicamente son, sin lugar a dudas, la mano articulada desarrollada por el DLR y la *Yaskawa Hand*, cuyas tecnologías de integración permiten desarrollar experimentos de buen nivel. Desgraciadamente, ninguna de estas es comercial.

requerimos de una mayor fidelidad en las interfaces: la restricción primera sigue siendo la integración multisensorial. En particular las posibilidades de integrar capacidades táctiles con aquellas visuales, demandan la integración de una plataforma donde ambas entidades puedan compartir un mismo escenario. Con el fin de conocer los principios de operación de los sistemas con retroalimentación de fuerza, se adquirió uno de los más baratos del mercado al cual se le procedió una dura cirugía. Lo más importante para nuestro grupo era el de mantener la capacidades mecánicas íntegras y proceder a interfasear dicha plataforma con nuestro sistema computacional embebido cuya única función es la de leer la información de los sensores de posición y transmitir las corrientes computadas al interior de nuestro sistema de cómputo: visualizar la información de los sensores de posición en movimientos coordinados del supuesto enlace virtual y efectuar el cómputo de la dinámica ocurrida como efecto de tal interacción. El principio fundamental de comportamiento háptico computacional es el de calcular las fuerzas de reacción del dispositivo mecánico a partir de una estimación del movimiento: relacionar dicho movimiento con la naturaleza de la acción y ponerla en correspondencia con los objetos tanto reales como virtuales que pertenecen a la escena. Transmitir las fuerzas de interacción parece ser uno de los puntos de mayor interés y en cuya naturaleza se originan gran parte de las limitaciones de los sistemas hápticos actuales. Cuando dicho cómputo se complica, como sería el caso de que el paso de integración numérico sea mucho mayor que los tiempos de respuesta del sistema - también conocido como *latencia* - se tendrán efectos de retardo en cuyo caso las percepciones otorgadas al operador serán muy distintas a las dictadas por la interpretación visual. Cualquier incoherencia detectada por el usuario en dichos términos puede provocar, además de frustración, mareos.

Cómputo de las fuerzas de reacción

Assumiendo a q como el vector que define la configuración articular de la plataforma mecánica, x como el vector de posición de la misma, tenemos

$$\dot{q} = J^{-1}(q)\dot{x}$$

donde J es el jacobiano del sistema. Para el cálculo de las fuerzas de reacción ideales τ y la fuerza f tenemos que

$$\tau = J^T f$$

En nuestro caso de estudio la matriz J es una matriz cuadrada, cuyas singularidades aparecen solo cuando el vector de fuerza está fuera del espacio nulo de la matriz. La dinámica general del sistema puede ser simulado acorde a la ecuación,

$$\tau + J^T f = M(q)\ddot{q} + C(q, \dot{q}) + N(q, \dot{q})$$

dónde las matrices M , C y N , representan las fuerzas de inercia, coriolis y gravedad o fricción. La solución aproximada de este sistema puede resolverse por integración numérica y de esta forma configurar los parámetros de respuesta del sistema. Debido a que los algoritmos existentes para la solución numérica están en orden de $O(n^2)$ para el sistema de dos grados de libertad es relativamente sencillo resolverlo con MAPLE o Matlab. La sensibilidad de la respuesta calculada puede ser analizada en términos directos de la matriz J , usando la convención del *matrix condition number*[26], tal que

$$\text{cond}(J) = \|J\| \cdot \|J^{-1}\|$$

esta condición es particularmente importante cuando no se cuenta con un sensor de fuerza, por lo que los valores de los torques τ se calculan como aproximaciones de

$$\tau = J^T \hat{f}$$

dónde el vector aproximado de f , denominado $\hat{f}$ es computado como $\det(q \cdot \dot{q}^T)$ aplicado en la dirección de $\dot{q}$. Secuencias de impulsos pueden aplicarse al sistema háptico a partir de

$$\min_n \sum_{i=1}^2 (\tau_i - g(t_i, f))^2$$

Este dispositivo fue construido a partir de un Joystick comercial, con retroalimentación de fuerza, el cual fue desarmado para poder reutilizar sus propiedades mecánicas. Dicho dispositivo está construido con dos motores de corriente directa fijados a la palanca de desplazamiento, los cuales nos sirven para dar la percepción al usuario de que se está teniendo contacto con los objetos del entorno virtual; al igual que estos se encuentran también fijados dos sensores de posición los cuales proporcionan valores de voltaje para poder determinar en qué posición se está empujando dicha palanca. Estos dispositivos son conectados a una interfaz construida con microcontroladores PIC la cual se encarga de establecer la comunicación serial con la computadora provista con el sistema operativo Linux en la cual se tiene un entorno gráfico donde se visualizan las interacciones. Este entorno consiste de un plano en el cual se desplaza un objeto a partir del movimiento en la palanca del dispositivo; además de este objeto, existe otro que se encuentra fijo y que es aquel con el cual colisionará el primero, provocando la reacción en el dispositivo háptico; todo este ambiente es construido en un software libre de simulación gráfica llamado *Geomview*, el cual ejecuta secuencias de instrucciones de un programa desarrollado en lenguaje C. La interfaz recibe los valores de voltaje de los sensores y los envía a la computadora como un número entero, esto es, a través de la comunicación serial establecida por estos, una vez que se ha enviado la lectura de los sensores, el programa que se ejecuta en la computadora, procede a realizar el análisis de los datos recibidos, mientras la interfaz espera una respuesta de la PC. Este análisis de datos consiste en determinar hacia dónde se debe de desplazar el objeto del entorno gráfico y detectar si se presentó una colisión con el otro objeto, y a partir de esto, informarle al software de simulación la nueva posición de los objetos para que los represente. La colisión es detectada a partir del cálculo de las distancias entre los objetos con la función de distancia entre dos puntos tomando en cuenta su punto central. Finalmente si se presentó la colisión el programa informa a la interfaz la dirección en la cual tuvo lugar para que activen los motores en sentido contrario para así dar la sensación de que se ha chocado con un objeto. El trabajo continuará evaluando el desempeño del dispositivo en términos cuantitativos; procederemos al estudio de propiedades dinámicas del operador en aspectos de sensibilidad de fuerza así como de percepción y fusión de datos. Nuestro interés último es el estudio de interfaces hápticas para la manipulación diestra, por lo que este primer trabajo da la pauta para el diseño de un dispositivo háptico para manipular objetos virtuales, en un principio y de objetos reales cuando hayamos finalizado la construcción de la mano articulada. Los autores ponen a disposición de la academia tanto el software como la experiencia en el desarrollo del hardware para aquellos interesados en fomentar el estudio computacional y el uso de las interfaces hápticas. La creación de una comunidad inter universitaria que pueda entrar en contacto háptico es uno de los intereses primordiales de este proyecto.

3.1.3. Bioinformática háptica⁹

Dada una proteína y un motif (un patrón estructural repetido) de esta misma, ubicados en posiciones iniciales, se realizará la correspondencia estructural (matching) de este motif dentro de la proteína, es decir, encontrar el lugar correspondiente del motif en la proteína. El proceso numérico de optimización genera mínimos locales por lo que se usa una interfaz háptica para conectar al usuario con el método numérico de forma interactiva. La búsqueda de un motif es típicamente hecha para identificar los sitios en los que la proteína interactuará con otras moléculas. Un método para encontrar un correspondencia para un patrón dado, tiene numerosas aplicaciones como, el protein ligand-binding, minería de datos en las bases de datos de las estructuras de proteínas. Avances significativos en estas tareas de bajo nivel facilitarán tareas de alto nivel como el diseño de drogas. El uso de sistemas hápticos para dicho proceso es un método de reciente interés en las ciencias computacionales. El proceso de encontrar una correspondencia consiste en tener una proteína y un segmento de ella (motif) para tratar de encontrar la posición de la proteína en la que el motif, se acopla. Este procedimiento consta de 2 pasos:

1. Determinar un subconjunto de puntos en la proteína cuyas distancias a los puntos del motif sea mínima. A este subconjunto de puntos se le llamara puntos ICP, por el algoritmo ICP que se mencionará mas adelante.
2. Encontrar un conjunto de transformaciones en el espacio (rotaciones y traslaciones) que acerquen los puntos del motif al subconjunto de puntos ICP.



Figura 12: Configuración inicial y final del motif buscado. Por medio del sistema háptico, el operador puede sentir el mínimo local numérico, como resultado del computo. El puede, a voluntad, ejercer una fuerza y sacar el algoritmo del mismo.

Inicialmente desarrollamos un programa el cuál consistía en aparear puntos en 3-D, dado un grupo de puntos P en R³ y un subconjunto S de estos. Se inicia el subconjunto S en una posición inicial lejana al conjunto P, para después aplicar este algoritmo y encontrar correspondencias de S en P y posteriormente las transformaciones necesarias (rotación y traslación) para lograr que S se alinee en el lugar que le corresponde dentro de P. El objetivo del programa es minimizar las distancias entre los puntos de S y de P y se habla de un error, el cual consiste en el promedio de las distancias de los puntos P y S. Se pretende que este error sea cero o que caiga dentro del margen de error establecido en un número de iteraciones dadas. El error es guardado para poder compararlo con el error de la iteración anterior y si la diferencia del error anterior con el actual es muy pequeño y el error actual es grande todavía, se considera que se está en un mínimo local. El programa acaba cuando el error esta

dentro del margen establecido o se ha alcanzado el máximo de iteraciones permitidas. Observamos que en algunos casos no se lograba minimizar el error lo suficiente debido a la posición inicial y en otros casos debido a que se alcanzaban algunos mínimos locales. Posteriormente usamos las coordenadas euclidianas de cada átomo de las proteínas, que se obtienen de archivos PDB (Protein Data Bank) y para poder visualizar que era lo que sucedía en cada iteración se decidió representar la estructura tridimensional de las proteínas. Las coordenadas son leídas de los archivos PDB y son guardados en una matriz dentro del programa y enviados a un software de gráficos tridimensionales (Geomview) para su visualización. Se usa una esfera para representar cada átomo en la proteína. Así todo lo que ocurre en el proceso de matching se puede visualizar. Las proteínas completas pueden tener más de 2000 átomos y la interfase, Geomview, no nos permite cargar más de 1500, cuando cargáramos más de 700 el programa se volvía muy lento, debido a que se requiere mucha memoria. Para un mejor desempeño optamos por trabajar con un archivo con 300 átomos para la proteína (modelo fijo), y el motif sería uno de 100 o menos. Este lo representamos con esferas rojas. En el nuevo algoritmo los átomos de la proteína sustituyen al conjunto de puntos P y el motif al conjunto S. Ahora podíamos observar claramente que era lo que sucedía al alcanzar un mínimo local, el motif quedaba oscilando en una región de la proteína y no podía salir de ahí. Cuando corremos el programa además de observar los movimientos del motif en el Geomview, también observamos en la consola los errores que se obtienen en cada iteración y al final del programa, el número total de iteraciones realizadas. Al intentar minimizar las distancias entre los puntos ICP y los puntos del motif surge el problema de llegar a un mínimo local. Un mínimo local hará que el programa no alcance el mínimo global y por lo tanto no se llegue al sitio de la proteína en el cual se realiza el verdadero matching. El método de los cuaterniones solo garantiza que el algoritmo convergerá a un mínimo local [], pero no garantiza que siempre se obtenga el mínimo global. Una vez detectado un mínimo local, al principio, aplicábamos una rotación aleatoria a los puntos del motif para que estos salgan de la posición que los hace estar en el mínimo local, sin embargo puede ocurrir que una vez hecha ese movimiento aleatorio el algoritmo vuelva a colocar los puntos del motif en ese mínimo local o en otro. Algunas vez funcionaba pero en otras no se lograba el matching. Para atacar este problema incluimos un componente humano para tratar de sacar el programa de un mínimo local. Este componente humano puede interactuar con el programa a través de una interfaz háptica. La interfaz háptica consta de un joystick con dos grados de libertad, que cuenta con dos motores. Le agregamos un switch para intercambiar entre rotaciones y traslaciones. Habíamos pensado en hacer este intercambio presionando una tecla, pero se uso el switch ya que nos comentó nuestro compañero que es mucho más fácil esto, ya que el uso del teclado sería complicado ya que se requiere de usar interrupciones debido al uso del Tower, que es un sistema modular de desarrollo para el diseño y prototipo de dispositivos computacionales, que es usado para trabajar con la computadora y el háptico. Habíamos pensado en usar otra tecla para poder intercambiar los planos y poder usar todos los movimientos pero debido a que hubiéramos tenido que usar otro switch preferimos no hacerlo ya que se complicaría el uso del háptico. Una vez que se ha identificado el mínimo local la interfaz háptica nos avisa por medio de vibraciones y además en la consola se detiene la impresión de los errores y se espera a que el usuario indique hacia donde quiere que rote o hacia donde quiere que se traslade para

⁹Esta trabajo es desarrollado por Patricia ortegón, Gerardo May y Eyra Woo. Durante su estancia en México, el Dr. Fabian Schwarzer nos apoyó en la introducción al tema así como el desarrollo del sistema. Ricardo Pérez estuvo a cargo de la puesta punto del sistema háptico

sacar el motif del mínimo local. Una vez que se realiza este movimiento el resultado se puede ver en el Geomview y el programa continúa con su función normal. Debemos regresar la palanca al centro por que si llega a otro mínimo local el programa lee la posición en que se encuentra la palanca y son los valores que aplica como rotación o traslación. Los diferentes valores que toma el error se van guardando en un archivo para poder graficarlos más tarde, lo mismo se hace con las componentes de la diagonal de la matriz de rotación y los elementos del vector de traslación. Graficar todos estos valores nos sirve para poder observar el comportamiento de los mínimos locales, que tan rápido converge el error a cero y así tratar de descubrir algún patrón que nos ayude a comprender mejor que es lo que ocurre con el proceso de matching. El programa fue probado con una proteína de 330 átomos y un motif de 100 átomos. El motif tiene una traslación inicial de 10 unidades en X, Y y Z con respecto a la proteína. En esta prueba llegamos a cuatro mínimos locales antes de llegar al mínimo global con un número de iteraciones igual a 66. En otra prueba con las mismas características iniciales alcanzamos el mínimo global pasando por cuatro mínimos locales en 59 iteraciones. Para sacar el motif de los mínimos locales usamos en los dos casos únicamente traslaciones. En otra prueba donde usamos la traslación y rotación no logramos llegar al mínimo global aunque nos quedamos muy cerca, esto se debe a que resulta muy difícil controlar las rotaciones con el háptico. Otro problema que notamos fue que en algunas ocasiones no se pudo reconocer algunos mínimos locales que se presentaban en forma de patrones repetidos en los errores, y el motif se quedó oscilando entre ellos, algunas veces eran tres movimientos y otras siete, los que repetía el motif hasta alcanzar el máximo de iteraciones. Notamos que sin una orientación de los ejes de coordenadas era muy difícil decidir cual era el movimiento adecuado para llegar a la posición correcta. Además de cargar las esferas cargamos en el Geomview unos ejes de coordenadas con los cuales tuvimos una mejor orientación sobre los movimientos que se realizan. Debemos aclarar que solo nos podemos mover en X y Y debido a que el háptico solo cuenta con dos grados de libertad. El problema que esto origina es que si en algún momento se llega a necesitar un movimiento sobre Z este no se podrá realizar, y nos es más difícil lograr el movimiento necesario solo con X y Y. Se observó que el motif siempre tiende a dirigirse hacia donde hay una mayor concentración de átomos aun cuando realizamos una traslación que lo aleja bastante siempre tiende a regresarse. En la grafica del error pudimos observar claramente donde son alcanzados los mínimos locales y el movimiento que realiza el joystick para sacarlo de ahí, ya que observamos cuatro picos bien definidos en la grafica. La grafica de los componentes del vector de traslación nos indica cuales son los movimientos realizados en el joystick y sobre que ejes fueron hechos, además los números de iteración en donde se presentan los picos en las graficas de traslación y de error coinciden. Otro experimento realizado fue comparar una proteína con ella misma y en este caso el número de iteraciones fue mucho menor debido a que no se pueden alcanzar muchos mínimos locales.

3.2. Ejemplo de Robótica y Visión por Computadora en Neurofisiología

Algunos de los problemas de modelado de conducta en ratas pueden ser estudiados usando técnicas computacionales, donde a través de la observación de sus movimientos, se pueden hacer análisis cuantitativos sobre su comportamiento cinemático. Identificar patrones de conducta es-

tereotipada puede parecer evidente a simple vista, previo un entrenamiento y concentración mínimos.

Cualquier incertidumbre es compensada con la experiencia del observador, al igual que patrones mal clasificados pueden aceptarse como buenos a consecuencia de lo mismo. La visión computacional ofrece el poder medir las condiciones cinemáticas y dinámicas de un sistema articulado complejo pudiendo formar así un esquema de patrones, donde la incertidumbre humana puede verse reducida a la definición de ciertas tolerancias. De esta forma, deberíamos ser capaces de determinar las diferencias conductuales de un roedor en presencia de fármacos; como por ejemplo, lograr determinar si el acicalamiento del roedor se incrementa o decrementa al incrementar la cantidad de fármaco administrado. El carácter multidisciplinario ha llevado al grupo de Instrumentación, Control, Visión y Robótica de la Facultad de Matemáticas de la Universidad Autónoma de Yucatán y al equipo del Laboratorio de Neurofisiología del Centro de Investigaciones Regionales "Hideo Noguchi" a emprender este proyecto¹⁰

3.2.1. Seguimiento del movimiento

Es fundamental permitir el libre movimiento del animal en observación pues su conducta será menos alterada por factores externos y manipulación amén de ser más fácilmente observable. Así mismo, es importante interferir al mínimo en el comportamiento experimental por lo que la visión por computadora ofrece las condiciones necesarias para llevar a cabo la observación. La identificación acertada de la configuración osea de la rata depende en gran parte de la visibilidad que de ella se tenga. Por lo mismo, el sistema de visión computacional cumple la misión inicial de localizar a la rata dentro de la caja y realizar el "zoom" correspondiente para focalizar en todo momento la anatomía de la rata. Por razones de precisión se considera la inserción de un "back-pack" coloreado y posicionado en la espalda de la rata, donde además se almacenará posteriormente el sistema transmisor de control de electrodos y registro de actividad eléctrico extracelular que se encontrará en la cabeza. El estudio con animales vivos no es práctica común fuera del laboratorio y muy pocas veces vemos animales de experimentación en centro de cómputo. Por lo que hemos procedido, inicialmente, en validez las metodologías de estudio sobre una plataforma de simulación que presente algunas de las características de la plataforma real.

3.2.2. Formulación del problema

Dadas las coordenadas $x_{caja} - y_{caja}$ de la posición de la rata, estimada a través de la Caja de Actividad Motora necesitamos referir el sistema de visión artificial con las cámaras de video hacia la zona de interés. Habrá que calcular $x_{vision} - y_{vision}$ sobre los cuales se estimará $q = (q_1, q_2, \dots, q_n)$ donde n es el número de grados de variables que definen la configuración anatómica de la rata. Se define C_i como la i -ava conducta, la cual se obtiene al integrar una secuencia de configuraciones $\sum f(q)$ donde f es la función de relación de la conducta C_i en la configuración q .

3.2.3. Generación de datos a partir de las imágenes virtuales

Proponemos dos métricas que nos sirven como parámetros para identificar la conducta correspondiente. EL primero

¹⁰Esta integración está siendo financiada por el convenio Sectorial SEP/CONACYT 2003-2006

hace referencia a una generación de familias de líneas $y = ax + b$ usando la transformada de Hough donde encontramos un espacio ab donde las regiones de conexión nos definen los parámetros de la conducta. Después, obtener el *convex hull* en el espacio de parámetros ab . Debido al origen de las líneas, el *convex hull* aglutina la incertidumbre en la posición de los miembros. El segundo espacio métrico corresponde a un análisis de densidad de negros frente a blancos con el fin de ubicar el centro de gravedad aproximado del modelo. Lo cual servirá de referencia para la ubicación del sistema.

3.2.4. Etapa 1: Generación de estímulos y cómputo de conductas artificiales

El sujeto en observación (rata) puede obedecer a conductas inducidas por estímulos. Dichos estímulos pueden ser generados a través de secuencias de luces o ruidos, produciendo así cambios en la configuración ósea o secuencias de movimientos. En el modelo sintético, nuestra rata virtual llevará a cabo movimientos cinemáticos generados de forma aleatoria. A partir de esa imagen, nosotros humanos podríamos decir que la rata dice "no" con el movimiento de la cabeza. Sin embargo en la simulación se generaron movimientos aleatorios en algunas de las articulaciones (en 7 para ser exactos, incluido un ligero movimiento de la cola) y cuyo movimiento total podría obedecer a una cambio de conducta más significativo. Ejemplo Parte I Un problema computacional interesante corresponde al tipo de distribución estadística necesaria para generar cambios de configuración, cuya comportamiento no se requiere sea uniforme. Si la función de distribución acumulada de la función de densidad se considera como invertible, podremos generar secuencias aleatorias a partir de números random e invertirlos. Un caso útil es la función de distribución exponencial con parámetro λ cuya función de densidad es $f(t) = \lambda e^{-\lambda t}$ cuya función de distribución acumulada es $F(x) = \int_0^x f(t) dt = 1 - e^{-\lambda x}$ [26]

3.2.5. Etapa 2: Identificación de patrones de movimiento estereotipados

En la Fig. 13 podemos observar un par de triadas de imágenes. Las de arriba representan a la misma configuración cinemática de la rata virtual. En la primera identificamos al modelo tridimensional. En la siguiente observamos el modelo volumétrico del animal y en el última vemos la sombra bidimensional del mismo. Es claro que sobre esta última es difícil saber e identificar la configuración del animal. El modelo volumétrico es el sistema de referencias sobre el cual la visión se referirá para la localización del sistema. Sobre esta última es sobre la que se definirán las conductas a identificar; así mismo la incertidumbre quedará limitada a ese espacio.

Ejemplo Parte 2 A partir de los patrones definidos conforme a la función de distribución exponencial, nos interesa identificar las conductas conforme a los datos generados, esto es: dado un $y = F(x)$ podemos resolver para x dado que

$$x = \frac{\log(1 - y)}{\lambda}$$

Así podemos *samplear* de la distribución exponencial tal que,

$$x_k = -\frac{\log(y_k)}{\lambda}$$

3.2.6. Etapa 3: Identificación de restricciones de temporalidad a las conductas

Los patrones identificados sobre la función de distribución seleccionada no nos dicen nada sobre la temporalidad, puesto que aumentaríamos la dimensión para el espacio de conductas generadas, como si fueran completamente independientes. En ingeniería mecánica se conoce como el *espacio de configuración* donde se pueden representar una configuración particular por medio de un punto en el espacio n -dimensional. Para cada uno de esos puntos podemos agregar algún sub-espacio de la función de probabilidad para la transición. Hay trabajos que representan esta naturaleza dinámica con sistema de Markov, donde las transiciones se ponderan con distintos valores y cuya bidireccionalidad queda establecida en términos también de probabilidad.

3.2.7. Fusión de datos extrínsecos e intrínsecos

La naturaleza tan diversa de los datos provenientes sobre la conducta de la rata establece un problema de correspondencia importante. Identificamos 3 fuentes distintas:

1. Un fármaco conocido es administrado al roedor F
2. Se registra la actividad cerebral durante la observación AC
3. Se observa el movimiento del roedor y se estiman las conductas estereotipadas CE_n

3.2.8. Formulación del problema

Formulamos tres escenarios distintos donde se ilustra la complementariedad de la integración sensorial.

Problema 1 : Cómputo de la Conducta Estereotipada (CE_n) :

Dado el fármaco F y la actividad registrada AC , hacer una estimación cualitativa de CE

Problema 2 : Cómputo de la actividad cerebral (AC)

Dado el fármaco F y el conjunto de conductas estereotipadas CE_n , hacer una estimación cualitativa de la actividad cerebral AC

Problema 3 : Cómputo de la actividad cerebral (F)

Dado un registro de actividad cerebral AC y una familia de conductas estereotipadas, identificar el fármaco suministrado al roedor

3.2.9. Caracterización de la conducta estereotipada

La determinación del número mínimo de parámetros necesarios para la definición precisa de una conducta diferenciable de otras parece ser el problema principal de la caracterización única. El instrumento principal para la determinación automática de la conducta es la visión por computadora, donde el análisis de las imágenes permite la generación de índices cuantitativos sobre los cuales la toma de decisiones soporta su justificación. Por razones de estudio, nos propusimos identificar la magnitud del problema usando imágenes virtuales del movimiento del animal observado. La configuración cinemática es generada de forma aleatoria y el objetivo consiste en determinarla a partir de la sola observación de las imágenes. La parametrización de Denavit y Hartenberg es la más utilizada para definir movimientos cuasi-estáticos.

3.3. Ejemplo de Aplicación en Arqueología

Las tecnologías de la información hacen posible una mayor integración y explotación de recursos. Por medio de la



Figura 13: Conductas estereotipadas basadas en imágenes. Las imágenes de la izquierda representan el modelo 3D de la rata virtual. Las imágenes centrales corresponden a un modelo volumétrico donde se incluye la incertidumbre en la posición de cada uno de los componentes anatómicos. Las imágenes de la derecha son las "sombras" del modelo. (los autores agradecen a Miguel Sandoval, Amelia Perez Ovando, Audre Chan Cambranes el habernos facilitado su modelo)

digitalización de documentos, fotografías, videos y demás acervo que pueda considerarse de trascendencia histórica y/o documental se hace posible su distribución masiva. En aspectos de investigación científica es cada vez más importante el acceso a la información en cuyo defecto se corre el riesgo de repetir o re-incidir en aspectos que pueden ser de suma importancia para los resultados de intercambio. Las labores de investigación deben también formar parte del acervo cultural de la materia en cuestión, en nuestro caso nos interesa divulgar, tanto la información fundamental sobre un sitio arqueológico así como mostrar los resultados de las investigaciones que han dado lugar a la majestuosidad recuperada por la restauración. La sola concepción de los documentos como productos electrónicos cambia la percepción sobre el uso de las máquinas. Un problema de tal magnitud solo puede ser abordado desde una perspectiva multidisciplinaria; y el trabajo no es evidente. Siguiendo el esquema propuesto por Morris,[27] resulta primordial la validación tecnológica así como evitar al máximo los cambios en el diseño. En este proyecto se involucra la interacción con materiales y objetos los cuales no siempre se encuentran dentro de un centro de cómputo por lo que las estrategias de trabajo son delicadas.

3.3.1. La naturaleza dinámica de una zona arqueológica

La primera impresión al entrar en contacto con una zona arqueológica es bastante ambigua. Por un lado, la majestuosidad arquitectónica que presenta un complejo cuya integridad estructural estuvo cubierta, en su casi totalidad, por cientos de años, y a cuya misma naturaleza fue la de soporitar, de forma parcial en la mayoría de los casos, el paso de los años, nos lleva a justificar la importancia tanto cultural como histórica de un lugar antes considerado como sagrado. Por el otro lado, tenemos el enorme esfuerzo que dedican arqueólogos, antropólogos, albañiles, barrenderos, arquitectos en la restauración del sitio y en cuya belleza se alcanza a distinguir, a veces con polémica, la mano del hombre moderno. Han sido los mismos arqueólogos los que han ido desenterrando, capa tras capa, donde se han ido encontrando distintas manifestaciones arquitectónicas: su misma intervención es parte del dinamismo propio a la naturaleza de un sitio arqueológico. Dicho proceso de restauración es difícilmente percibido por el visitante; más aún, se convierte en parte de la experiencia ver que, por un lado, se está en continua investigación sobre el sitio y por el otro la canti-

dad de montículos que restan por descubrir, por el otro. Las tecnologías de la información han ido asumiendo un rol determinante en la economía mundial. Es casi regla general la imposibilidad de establecer un negocio exitoso sin la dependencia de las tecnologías computacionales. La investigación científica, al depender en gran parte del financiamiento para su ejecución, está sujeta a presiones competitivas al demandar una continuidad de innovaciones en los métodos y procesos que se involucran en su naturaleza. Poner al servicio de la comunidad un sitio arqueológico *digital* con la mayor velocidad y precisión dicta de alguna forma la derrama económica que sobre los sitios aledaños puede ocurrir. Creemos que un proyecto de cultura digital, debido a su carácter de fácil replicación, puede tener un impacto cultural y económico importante para una región donde existen aún 2070 sitios arqueológicos por restaurar.

3.3.2. Registro y anastilosis

Una de las alternativas para asegurar la correcta restauración de un montículo arqueológico es el de seguir la traza del derrumbe. A primera instancia un montículo virgen alberga todas y cada una de las piedras que originalmente lo formaron. Con el paso del tiempo, algunos de estos componentes han desaparecido como resultado de saqueos y otras vicisitudes. Bajo situaciones ideales, podemos limpiar el montículo de yerba y tierra que se ha ido acumulando con el paso de los años e ir registrando la posición de las piedras hasta poder armar, con cierta precisión un esquema del derrumbe. Nuestro interés es justamente el problema inverso para saber cómo estaban organizadas las piedras originalmente. Seguir el proceso de caída de las piedras a través de un registro se conoce como *anastilosis*.

Cómputo del registro

El arqueólogo toma fotos desde una posición libre de obstáculos. Tradicionalmente el registro de la imagen consiste en calcar artísticamente los bordes de los objetos diferenciados y calcarlos en papel especial y eliminar todo aquello ajeno a la naturaleza arquitectónica de la imagen, empleando gran cantidad de tiempo y esfuerzo. El procedimiento propuesto consiste en obtener una fotografía digitalizada del montículo. Computacionalmente hablando, la umbralización de la imagen, es decir transformar la imagen a tonos blancos y negros, y posteriormente se lleva a cabo el proceso de erosión y dilatación para reducir el ruido en la imagen. Enseguida se procede a clasificar las piedras por su tipo; contarlas, etiquetarlas y crear un aproximación de su profundidad con base a la calibración de la cámara.

Cómputo de la anastilosis

Con base en la información obtenida en el cómputo del registro, procederemos a obtener una aproximación de la anastilosis a partir del empleo de cadenas cinemáticas y la manipulación de los parámetros de la matriz de Denavit-Hartenberg, con la cual se pretende determinar la trayectoria del derrumbe de cada una de las piedras, de manera que al aplicar este método a la inversa obtengamos la trayectoria para la restauración del monumento arquitectónico. Para realizar lo anteriormente mencionado se utilizará software de cálculo matemático y de simulación en 3D.

3.3.3. La cerámica digital

Al ir explorando los montículos, los arqueólogos encuentran utensilios y herramientas, rasgos culturales que permiten datar la antigüedad del lugar. Uno de los componentes más significativos son todas las vasijas y utensilios que conforman la cerámica del lugar. Muy pocas de ellas son en-

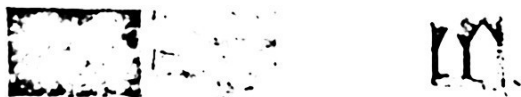


Figura 14: Imagen ejemplo del registro: qué tipo de tierra se puede localizar, cuántas piedras localizamos, qué tipo de piedras distinguimos y cómo podemos apoyar a la construcción automática. La primera imagen es una fotografía fiel del sitio. La segunda es la imagen de registro obtenida por un artista. La tercera es una imagen obtenida automáticamente. La última corresponde al montículo estimado.

contradas de forma integral, la mayor parte son obtenidas en pedazos. Uno de nuestros objetivos es desarrollar herramientas para la restauración de las vasijas a partir de una simple identificación de características. Previa digitalización y modelado de las piezas, un sistema de cómputo lleva a cabo la correspondencia entre las piezas planteando así hipótesis sobre la posible pertenencia de los trozos o pedazos a alguno de los modelos previamente digitalizados.

3.3.4. Documentación digital

Un proyecto integral sobre un sitio arqueológico debe, creemos, contemplar aspectos no solo históricos del lugar sino ponderar adecuadamente el carácter dinámico en el que se está inmerso. En la actualidad contamos con una gran cantidad de medios sobre los cuales se puede reproducir la información recabada durante la investigación de un sitio de esta naturaleza. Una vez recaba la información y digitalizada podemos proceder a adaptarla al medio más idóneo para su difusión. Con el fin de poder instrumentar el proyecto con personalidad propia hemos decidido desarrollar una investigación documental y mediática sobre los diversos aspectos, los cuales serán integrados en el sitio Web. Nos referimos a información histórica, aspectos culinarios, tradiciones etc. Estos aspectos forman el núcleo sobre el cual se integrará el sitio Web, el Tutorial/multimedia sobre el sitio arqueológico y el conjunto de Actividades para niños.

Página electrónica

La irrupción de Internet ha permitido el intercambio de información entre distintas entidades e individuos. En aspectos culturales han habido intentos serios sobre la redefinición de los medios de comunicación electrónicos, desgraciadamente la banalización en el uso de dicha tecnología ha creado actitudes negativas frente a la inmensidad de información irrelevante e innecesaria sobre aspectos de mayor importancia. Transmitir la grandeza de un sitio arqueológico transportando la información de papel a medios electrónicos puede ser un desastre. En la actualidad no existe un sitio en la red que contenga además de información útil al posible visitante, información relevante a la dinámica de la zona arqueológica. Se presenta una visión general sobre la utilización de las tecnologías de la información para la creación de un proyecto de cultura digital sobre un sitio arqueológico. El problema computacional es de índole cultural debido a que el acervo sobre el que se fundamenta la información forma parte de la cultura del lugar. Algunos componentes no son automáticamente digitalizables, por lo que es necesario incursionar en procesamiento digital de imágenes y visión por computadora. Esperamos que este documento sirva de base para la concepción computacional y tecnológica sobre la cual los mecanismos modernos de divulgación y pedagogía deben incursionar.

3.4. Ejemplo de aplicación de la robótica en nanomanipulación cuántica

La mayoría de nosotros hemos usado un ascensor. Solemos encontrar compañía al descubrir nuestro rostro reflejado en el espejo. Más de uno de nosotros nos hemos divertido viendo como nuestra imagen se refleja al infinito rebotando de lado a lado. Siendo un poco observadores, podemos percibir como la imagen va reduciéndose en tamaño, al igual que empieza a haber un retraso en el reflejo, conforme nuestra imagen se va perfilando a desaparecer en el infinito. Si tuviéramos la paciencia de sacar una pequeña moneda de nuestra bolsa, colocarla entre nuestros dedos y moverla con lentitud, podríamos observar paso a paso a través de las imágenes el proceso de manipulación. Si tuviéramos la posibilidad de registrar todas las imágenes generadas - hasta la más lejana - donde la moneda tuviera el tamaño de un átomo, seguramente el comportamiento no sería tan predecible ni parecido a lo que vemos a simple vista. A escala atómica, la mecánica que rige la relación de las fuerzas y el movimiento, deja de obedecer a los principios que estamos acostumbrados: aquellos que son perfectamente descritos por los principios de la mecánica clásica. De hecho, la misma luz que nos permite reflejar cada una de las imágenes, afecta el movimiento de la moneda a partir de imágenes de un cierto tamaño. Viendo en detalle el movimiento de la moneda, que se origina como resultado de las fuerzas que son transmitidas por las yemas de nuestros dedos, sería como pensar qué, estando la moneda flotando libremente en el aire, le pegáramos unos hilos y de ahí, la jaláramos en distintas direcciones. En las fiestas infantiles se da un ejemplo similar. Al jugar con una piñata tratando de evitar que la rompan por lo general, la hacemos moverse de arriba abajo, jalando de las cuerdas que la sostienen. Si se tira de ambas, las cuerdas se tensan y la piñata tiende a subir. Si una de ellas se suelta un poco, la piñata baja. Al igual que una moneda es manipulada por la mano. Si consideramos aquella imagen donde la mencionada mecánica clásica pierde validez, veríamos una moneda muy pequeña y la racionalidad sobre su comportamiento nos obligaría a pensar de manera distinta. Las fuerzas que se aplican lo hacen de manera digamos, controlada, de otra forma esta tendería a escaparse de los dedos. Lo que implica el considerar un mecanismo de control de las fuerzas. Este mecanismo se da normalmente, después de un largo proceso de aprendizaje a lo largo del primer año de vida en los humanos, pudiendo a la edad de 1 año y meses poder, por ejemplo, insertar dos piezas de lego. Esta tarea, por simple que se conciba, requiere el poder combinar tanto el correcto posicionamiento de las piezas, como un mínimo de coordinación de las fuerzas que se aplican. De otra manera, la tarea sería imposible de realizarse. Recordemos que nuestro tema en cuestión es el de manipular objetos sumamente pequeños. La experiencia descrita anteriormente puede ejemplificar perfectamente un tipo de aplicación de la tele-manipulación, tanto al nivel microscópico - perceptible al ojo humano - como a nivel nanoscópico o como es nuestro caso de interés, nanoscópico. Veamos. Llamaremos sistema de tele-manipulación a unión de dos sistemas: el sistema *maestro* y el sistema *esclavo*. Nuestra mano corresponde - obviamente - al elemento denominado sistema *maestro* puesto que es quien genera movimiento; ya quella imagen remota, de tamaño ínfimo, denominamos sistema *esclavo*. Tendríamos así, los dos elementos mínimos necesarios para formar un sistema *perfecto* de tele-manipulación. Otros ejemplos ilustrativos que corresponden a las aplicaciones de la robótica como instrumentos de tele-manipulación, tanto a escala macro o micro se ilustran a continuación. Al nivel microscópico un buen ejemplo

sería el bien conocido por la industria nuclear, donde se utilizan robots esclavos para manipular objetos radioactivos u objetos muy pesados. A micro-escala sabemos ya de aplicaciones médicas donde los laparoscopios son instrumentados con micro-pinzas o micro-cámaras. Las diferencias entre nuestro sistema *perfecto* y los ejemplos descritos anteriormente difieren en que nuestro sistema, a través de los reflejos en los espejos, si bien *retarda* la imagen, es una copia fiel de lo realizado por la nuestra mano. En robótica, donde el sistema *maestro* y el *esclavo* pueden diferir entre lo que se manda ejecutar y lo que en realidad se ejecuta, esta copia no es tan fiel. Sin embargo, imaginemos que un tal sistema pudiera ser concebido. Si bien, podemos asumir como sistema *maestro*, una mano humana, nuestro sistema *esclavo* debiera ser un sistema electro-mecánico - o mecano-electrónico: mecatrónico por simplicidad- artificial realizable; más propiamente dicho, un robot. Dentro de todas las incertidumbres que gestan el área de la robótica, una de sus principales fuentes tiene origen al momento de planificar los movimientos. Ningún sistema de control automático, puede por definición, guiar la trayectoria de un robot perfectamente a lo largo de un camino predeterminado. Lo que hace necesario el contar con un ciclo de planificación que funcione a la par de los ciclos de control. La interacción de múltiples robots demandan de tareas más complicadas. Este es el caso de manipulación diestra, en el que se pretende controlar el movimiento de un objeto por medio de la interacción de un sistema multi-articulado. Por medio del control de los contactos, podemos manipular el objeto sobre la trayectoria deseada. En la cuasi totalidad de las aplicaciones industriales de la robótica distinguimos una enorme versatilidad en términos de movilidad estructural [28]. No así en el caso de las capacidades de manipulación fina de objetos. La razón principal de esta limitante se debe, principalmente, a que la mayoría de los robots industriales solo tienen la capacidad de tomar y transportar objetos. Son obvios los alcances que se pueden lograr al aumentar de forma sustantiva el potencial de manipulación, en particular, la concepción de sistemas autónomos para telerobótica [24].



Figura 15: Mano articulada con 12 grados de libertad desarrollada en nuestro laboratorio

3.4.1. La necesidad de una plataforma de experimentación abierta

Se han desarrollado más de 30 modelos distintos de manos articuladas. Algunas siguen el modelo antropomórfico en su diseño. Otras son actuadas con tendones y poleas. Algunas poseen características minimalíticas (e.g. reducir al máximo el número de grados de libertad) [29]. Sin embargo, todos y cada uno de estos proyectos se enfrentan en algún momento de su existencia frente a la necesidad de validar las características operativas del diseño. Es decir, si bien la mano puede parecer muy real, qué tan eficiente es para manipular objetos, objetivo inicial de cada proyecto. Si bien la manipulación diestra puede no ser el objetivo central, frente a la necesidad primera de tomar el objeto, ésta delimita el alcance y los objetivos de justamente, cómo tomarlo. Esta ac-

ción se conoce por los fisiólogos como la pre-prehensión (*pre-grasping*). Podemos nosotros hacer el experimento al simplemente tener la intención de tomar algún objeto cercano. Podemos ver como la *conformación* espacial de nuestros dedos se va acoplando en la medida en que nos acercamos al objeto. La manera en que configuramos la mano depende de la *intención* [30, 31] que tengamos al tomar el objeto. En general se han reconocido dos tipos de pre-prehensiones: aquella denominada *power grasp* que hace referencia a una acción en donde lo que nos interesa es transportar un objeto y no tirarlo durante su movimiento; y el otro conocido como *finger-tip grasp* donde el objetivo posterior a la prehensión es explícitamente el de manipular el objeto. De esta forma, la manipulación diestra asume el hecho de que el objeto está ya prehensado y el tipo de prehensión de tipo *finger-tip grasp*. De esta forma se procede a definir la estrategia de movimiento de los dedos a partir del movimiento deseado del objeto. El algoritmo diseñado para llevar a cabo esta tarea se designa *algoritmo para la planificación de movimientos para la manipulación diestra* y es un problema complejo. El que se ilustrará en la siguiente sección fue concebido en 1995 [21] e implementado en dos plataformas distintas [22]. Nuestro interés en la presente investigación fue el de crear las herramientas necesarias para realizar experimentos de forma dinámica en la concepción de distintas alternativas de algoritmos de esta naturaleza. La investigación en el diseño y concepción de manos articuladas no tiene más de 20 años por lo que es una área muy joven y que por motivos de pertinencia tecnológica es cada vez más evidente la necesidad de contar con herramientas computacionales sofisticadas para realizar experimentos de simulación, previos a la construcción de sistemas mecánicos. Seguirán nuestros colegas de ingeniería mecánica diseñando actuadores eficientes así como nuestros colegas de ingeniería eléctrica sistemas sensoriales muy sofisticados, pero hasta que nosotros, computólogos no logremos consolidar las herramientas necesarias para *fusionar* toda la información proveniente de sensores y generar las decisiones transmitidas en forma de corrientes compensatorias a los motores, no podremos avanzar en la comprensión de los mecanismos de manipulación diestra. A su vez un tal desarrollo requiere la fácil incorporación de información visual sobre la cual se contempla la posibilidad de conocer con cierta precisión la posición y orientación del objeto manipulado [23]. Es necesario pues, diseñar sistemas que soporten estas expectativas y que permitan tanto a expertos en la materia como a aquellos que se inician, contar con lo necesario para investigar esta tan importante tarea. Qué mejor que hacerlo sobre una plataforma estandar como el MAPLE. Indudablemente que mientras más accesible y disponible este un trabajo de investigación, en este caso, hablando específicamente de investigación en computación, más fácil será que los investigadores en el tema de que se trate, lo mejoren, lo enriquezcan, lo critiquen, lo valoren o en su caso, lo refuten y lo rechacen. Creemos que el MAPLE, es una herramienta, con características que lo hacen sobresalir para usarlo en el área de la graficación y/o de la simulación. Posee funciones pre-construidas en diversos campos de la matemática, y posee una capacidad de graficación bastante versátil. El producto está ampliamente difundido en el ámbito académico y está disponible para usar en varias plataformas. Una de las posibilidades más interesantes de MAPLE es la de generar archivos de código fuente en C (y/o Fortran) para evaluar las expresiones resultantes. Esto se puede hacer utilizando el menú contextual o introduciendo órdenes por el teclado. MAPLE tiene también grandes capacidades gráficas, en algunos aspectos superiores a las de paquetes como MATLAB. Permite hacer animaciones en dos y tres dimensiones.

Consideremos que una mano compuesta de tres dedos M trabajando en el espacio euclidiano $\mathcal{W}$, es representado por la unión de tres dedos $F_1 \cup F_2 \cup F_3$. En nuestro caso, cada dedo tiene tres grados de libertad para el cual definimos una configuración articular como la tupla $q = (\theta_1, \dots, \theta_6)$. El objeto a manipular, considerado como polihédrico se denota como O , cuya posición y orientación se determinan con el referencial T el cual es colocado por simplicidad en el centro de gravedad del mismo. Dada una configuración inicial y final definidos como T_0 y T_f respectivamente y la configuración inicial de la mano M , definida como q_0 . El problema denominado *manipulación diestra* consiste en encontrar una secuencia continua de configuraciones de la mano $\tau = q_0, q_1, \dots, q_n$, que lleven O desde T_0 hasta T_f . La selección de los q_i 's debe respetar las restricciones cinemáticas de los dedos así como la estabilidad de la prehensión.

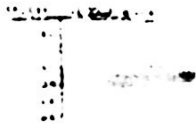


Figura 16: Simulación obtenida sobre MAPLE

Cabe aclarar que la trayectoria a seguir por el objeto se presume continua, por lo que los pasos de discretización conforme más pequeños aseguran una mejor y mayor precisión.

Sea $\vec{P}$ un vector que define la posición del contacto en la superficie del objeto. Sea $\vec{N}$ un vector normal al contacto en la superficie del objeto. Sea d la distancia euclidiana entre el vector de posición $\vec{P}_{\text{ant}}$ y el nuevo vector de posición $\vec{P}_{\text{new}}$ obtenido después de la transformación T_{new} . Sea d_e la distancia recorrida por el desplazamiento del punto de contacto sobre el punto original φ_0 y el candidato φ_i después de una interacción de búsqueda. τ_{nm} son los puntos a discretizar alrededor del punto original de contacto en el objeto (representado por el vector normal $\vec{N}_0$ y el vector de posición $\vec{P}_0$) y la superficie del dedo (representado por φ_0). El algoritmo se divide en 2 partes. En la primera parte se explora el espacio de soluciones potenciales para el problema de cambiar la orientación de un objeto mediante manipulación diestra. Este espacio es organizado como un conjunto de pequeños movimientos que pasan de una configuración a otra de la mano para cubrir la trayectoria deseada del objeto. La segunda parte consiste en la evaluación y selección de la mejor solución. Una vez aplicado el algoritmo obtenemos una secuencia de *modos* de manipulación para cada dedo. Al aplicar el algoritmo se obtiene una secuencia de posibles modos de manipulación (punto-fijo, rodamiento, deslizamiento) para cada dedo. Estas secuencias se ordenan de manera descendente buscando minimizar el desplazamiento angular de las articulaciones q .

3.4.2. Implementación computacional y convenciones

El código está dividido en tres grandes secciones: parámetros, procedimientos y algoritmo. En el primer bloque se definieron todos aquellos valores que son necesarios para iniciar con el algoritmo: la configuración de la mano articulada (número de dedos, posición y dimensiones), la configuración del objeto (posición, orientación y dimensiones), la configuración inicial de contacto entre los dedos y el objeto (puntos, ángulos). También se asignan valores

a los parámetros de discretización, es decir, aquellos datos que me permitirán buscar los puntos de contacto en el objeto y en los dedos, durante toda la trayectoria. Estos valores los toma de un archivo para posteriormente realizar la asignación. Cabe mencionar que los parámetros mencionados en el párrafo anterior, podrían ser también instanciados leyendo de un archivo. Hemos desarrollado una herramienta computacional para el estudio de la manipulación diestra. Debido a la flexibilidad del MAPLE es posible probar tanto estrategias de movimiento en un lenguaje de alto nivel en el cual el programa base del planificador está implementado. Este programa toma como base para los ejemplos un poliedro, y de esta forma se basa en planos y sus vértices para algunos controles. El algoritmo hace referencia a vectores normales en el punto de contacto, mas no menciona como deben ser calculadas, ya que eso va por parte de la persona que lo implemente. Calculamos las normales con una multiplicación vectorial, basado en la limitante señalada en el párrafo anterior, es decir, la utilización de objetos formados por planos. Las últimas sentencias del algoritmo hablan sobre el cálculo de fuerzas (torques) y que para seleccionar las tripletas de movimientos, se debe minimizar tanto el desplazamiento angular como las fuerzas necesarias para mantener la estabilidad. La implementación solamente está utilizando el desplazamiento angular como elemento para la selección de las tripletas. El programa puede ser enriquecido por medio de:

1. *Almacenamiento de los modos calculados.* El programa solamente hace un despliegue al momento de ejecutarse, no sería de mucha dificultad, tener la opción de almacenar estos resultados.
2. *Filtrado de modos.* Existen movimientos que aunque en el programa son detectados como los que provocan un menor desplazamiento angular, físicamente son imposibles de realizar, como por ejemplo: tres deslizamientos, o deslizamientos de los dos dedos superior y rodamiento del inferior, etc. Poder ir hacia atrás. Añadirle la propiedad de que si en algún punto de la trayectoria se presenta el caso de que ya no haya solución a la cinemática, poder regresarse a los pasos anteriores y seleccionar otra opción diferente a la que se eligió.
3. *Graficación opcional* El programa muestra la animación de la manipulación, solamente al final, se le puede incluir que también lo muestra en cada paso, sin embargo, se deberá analizar concienzudamente ya que se recargaría la memoria de la máquina. En la actualidad nos encontramos desarrollando una mano articulada donde comenzaremos a probar los resultados de las modificaciones actuales al algoritmo usando la plataforma presentada en este trabajo.

Algunos resultados experimentales de la implementación del algoritmo Dex-Pla en distintas plataformas robotizadas nos mostraron resultados prometedores en la manipulación de objetos. Sin embargo, el planificador requiere cierta precisión en la replanificación a manera de evitar la acumulación de errores. Debido a esto hemos incursionado en el uso de retroalimentación visual para la manipulación diestra y algunos resultados se muestran en Muñoz[23]. El trabajo futuro dentro de esta línea es la de considerar el mismo esquema de investigación en MAPLE y desarrollar las aplicaciones necesarias para realizar los experimentos y los algoritmos de control visual sobre esta plataforma.

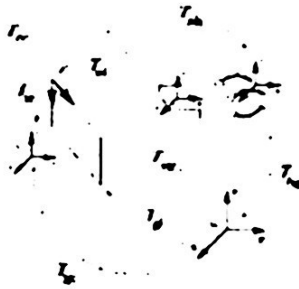


Figura 17: Perspectivas de integración visuomotriz para la manipulación diestra

4. Conclusión

Lo aquí presentado forma la primera aproximación hacia la elaboración de una cartera de proyectos sobre los cuales podemos ilustrar tanto los alcances de las ciencias computacionales como los problemas matemáticos involucrados. Dentro del cuerpo académico concerniente a las actividades de este grupo de investigación se cubren las materias de metodología de la investigación, la computación científica, las gráficas y la visión por computadora. Es importante recalcar el contexto académico sobre el cual se desenvuelven las actividades pues son, en la mayoría estudiantes de licenciatura los que participan en dichas actividades. Hemos podido constatar que dentro de la filosofía de *Rapid Prototyping* los alumnos muestran disposición para elaborar sus propias herramientas. Esto ha permitido cierta independencia tecnológica sobre los alcances de los proyectos. En particular sobre lo que corresponde a la programación sobre plataformas de desarrollo en Linux. Estamos empezando el programa de maestría en Ciencias Matemáticas donde pretendemos abordar problemas más específicos en el cómputo de sistemas complejos y visión por computadora.

5. Agradecimientos

Los autores agradecen a la Facultad de Matemáticas y a la Universidad Autónoma de Yucatán el apoyo otorgado a través del Proyecto PRIORI y a la SEP a través de PROMEP, lo que ha permitido la adquisición de la mayoría del equipo adquirido. A Bakhtiar Mikhak del Media-Lab quién a través del Gobierno de Puebla y el Centro de Tecnología Digital de Telmex nos ha permitido el uso del *Tower*. A Daniel Kornhauser quién ha aportado invaluable material al grupo. A Jean-Claude Latombe por el financiamiento de la visita a Stanford durante el verano del 2002 dónde nos iniciamos en la bioinformática. Al arqueólogo José Huchim del INAH, Mérida por la confianza en involucrarnos en sus tan importantes investigaciones. Al M. en C. Jorge Gómez el cuál sin su apoyo nos hubiera sido imposible incursionar en el diseño del proyecto de Cultura Digital sobre Uxmal. Al M. en C. Francisco Heredia quién su entusiasmo y confianza nos ha permitido involucrarnos a sus dominios de una manera tan agradable. A Fabian Schwarzer quién durante su visita nos apoyó en la creación del grupo de Bioinformática. A Otilio Santos, Víctor Uc Cetina, Carlos Aguilar, Fernando Curi.

Referencias

- [1] Cameron S. Enhancing gik. Computing minimum and penetration distances between convex polyhedra. In *Proc IEEE Int. Conf. On Robotics and Automation*, pages 3112 - 3117, 1997.
- [2] Lin M. C. *Efficient Collision Detection for Animation and Robotics*. Thesis for doctor of philosophy in electrical engineering at computer science, University of California at Berkeley, 1993.
- [3] M. L. Gómez Darrios. *Uso de Cilindros Generados para la Detección de Colisiones en Robótica*. Maestría en ciencias con especialidad en ingeniería en sistemas computacionales, Universidad de las Américas, Puebla, 2000.
- [4] Lin M. C. and Canny J. F. A fast algorithm for incremental distance computation. In *Proc IEEE Int. Conf. on Robotics and Automation*, pages 1008-1014, 1991.
- [5] Gilbert E. O., Johnson D. W., and Keerthi S. S. A fast procedure for computing the distance between complex objects in three dimensional space. In *IEEE Journal of Robotics and Automation*, pages 193-203, 1988.
- [6] O. Van Den Bergen. A fast and robust gik implementation for collision detection of convex objects. *Journal of Graphics Tools*, pages 7-23, 1999.
- [7] O. Van Den Bergen. Proximity queries and penetration depth computation on 3d games objects. *Game Developers Conference*, 2001.
- [8] Baris O. and Wong S. K. Image-based techniques in a hybrid collision detector. In *Proc IEEE Int. Conf. On Visualization and Computer Graphics*, pages 254 - 271, 2003.
- [9] R. Howard and O. Monga. *Visión por ordenador*. Paris, Editions Hermès, 1993.
- [10] G. Hirsinger P. Wunach. Registration of cad-models to images by iterative inverse perspective matching. In *13 International Conference of Pattern Recognition*, pages pp 77-83, Vienna, Austria, Agosto 1996.
- [11] N. D. MacKey P. J. Deseul. A method for registration of 3d shapes. In *IEEE Trans. Pattern Analysis and Machine Intelligence*, pages 14(2) 239-256, Febrero 1992.
- [12] R. M. Haralick and H. Joo. Pose estimation from corresponding point data. In *IEEE Trans. Systems, Man, and Cybernetics*, pages 18(6) 1426-1446, Noviembre 1989.
- [13] P. V. C. Hough. A method and means for recognising complex patterns. U.S. patent 3,069,854, 1962.
- [14] J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6) 679-698, 1986.
- [15] D. O. Lowe. Robust model-based motion tracking through the integration of search and estimation. *International Journal of Computer Vision*, pages pp 441-460, Agosto 1992.
- [16] Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *Image and Vision Computing Journal*, Vol. 15(No. 1) pp. 89-76, 1995.
- [17] Miguel Estrada-Arias and A. Lererf. A real-time fpga architecture for computer vision. *Journal of Electronic Imaging*(SPIE-IS-T), 2001.
- [18] Kazuo Tanie. *Handbook of Industrial Robotics*, chapter Robot Hands and end-effectors. John Wiley and Sons, 1999.
- [19] Thomas Lozano-Perez and Emmanuel Mazer. *Handey: A Robot Task Planner*. The MIT, 1992.
- [20] Alan M. Wing, Patrick Haggard, and J. Randall Flanagan. *Hand and Brain*. 1998.
- [21] L. A. Muñoz, Ch. Bard, and J. Sajera. Dexterous manipulation: A geometrical reasoning point of view. In *Int. Conference of Robotics and Automation*, Nagoya, Japan, May 1993.
- [22] L. A. Muñoz and Thomas Doornik. *Advances in Robotics*, chapter Simple Controllers for Dexterous Hands, pages 77-96. World Scientific Publisher, 1996.
- [23] L. A. Muñoz. Robust dexterous manipulation: A methodology using visual servoing. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Victoria(CA), October 1998.
- [24] R. L. Thompson, Iain Reid, L. A. Muñoz, and D. W. Murray. Providing synthetic views for teleoperation using visual pose tracking in multiple cameras. *IEEE Transactions on Systems, Man and Cybernetics*, 31(1) 43-54, 2001.
- [25] T. Masie and K. Salisbury. The phantom haptic interface: A device for probing virtual objects. In *ASME Winter Annual Meeting*, 1994.
- [26] Michael T. Heath. *Scientific Computing: an introductory survey*. Mc Graw Hill, 2001.
- [27] Peter Morris. *The Management of Projects*. 1996.
- [28] Shymon Y. Nof. *Handbook of Industrial Robotics*. Wiley, 1998.
- [29] A. Bicchi, A. Marigo, and D. Prattichizzo. Robotic dexterity via nonholonomy. In Springer Verlag, editor, *Workshop on Control Problems in Robotics and Automation*, CDC'97.
- [30] I. Kitagishi Y. Hayakawa and S. Sugano. Human intention based support trigger from "work triangle". In *Proc IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 930-935, Victoria BC, Canada, 1998.
- [31] Yoshinori Kuroo, Tomoyuki Ishiyama, Satoru Nakaniishi, and Yoshiaki Shirai. Combining observations of intentional and unintentional behaviours for human-computer interaction. In *Proc. on Computer Human Interaction*, pages 238-243, 1999.