Aportes en la Generalización de Habilidades en Aprendizaje por Imitación de Robots

José Gabriel Hoyos Gutiérrez

Universidad Nacional de Colombia
Facultad de Ingeniería y Arquitectura, Doctorado en Ingeniería-Automática
Manizales, Colombia
2015

Aportes en la Generalización de Habilidades en Aprendizaje por Imitación de Robots

José Gabriel Hoyos Gutiérrez

Tesis presentada como requisito parcial para optar al título de:
Doctor en Ingeniería-Automática

Director:
Flavio Prieto

Línea de Investigación:
Robótica
Grupo de Investigación:
GAUNAL

Jurados:
Eduardo Morales Manzanares, INAOE, México
Ricardo Ramírez Heredia, Universidad Nacional, Sede Bogotá
Sandra Esperanza Nope Rodríguez, Universidad del Valle

Universidad Nacional de Colombia
Facultad de Ingeniería y Arquitectura, Doctorado en Ingeniería-Automática
Manizales, Colombia
2015

Agradecimientos

A los profesores Flavio Prieto y Guillem Alenya les agradezco por todo el apoyo que me dieron para la realización de este trabajo.

Al Instituto de Robótica e Informática Industrial (IRI) del CSIC-UPC en Cataluña y al Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE) de Mexico, por su asesoría y facilitación de los equipos.

A las siguientes instituciones que apoyaron con recursos económicos esta investigación:

- Universidad del Quindío.
- Universidad Nacional de Colombia-Sedes Manizales y Bogotá.
- Colciencias-CONACYT Proyecto movilidad No. 533.
- Universidad de Guadalajara en Mexico.

Resumen

En programación por demostración (PpD) de robots, las variaciones de posición de los objetos relacionados con una tarea, requieren nuevas trayectorias que respondan a estas. Una de las técnicas existentes, es el modelo de mezcla de gaussianas parametrizado en la tarea. Este modelo permite relacionar los movimientos del robot con metas y poses de objetos, los cuales son los llamados parámetros de la tarea. Un problema que aparece es que se deben generalizar las trayectorias tanto en el espacio cartesiano, como en el de articulación, más específicamente, se requiere tener la cinemática inversa del robot, con el cual se puedan estimar las trayectorias de articulación, a partir de las trayectorias cartesianas. Un segundo problema que se presenta cuando se manejan objetos deformables, es que se pueden presentar fallos de ejecución, lo que requiere de una o varias acciones de recuperación. Un tercer problema es que, aunque las técnicas de generalización responden ante cambios, en ciertas ocasiones es necesario incluir nuevos comportamientos, los cuales pueden ser diferentes a los ya aprendidos. Este trabajo, se centra en tres aportes relacionados con generalización de trayectorias: *i)* El aprendizaje y aplicación en PpD de la cinemática directa con una red neuronal llamada máquina de aprendizaje extremo y con la cual se estima la cinemática inversa; *ii)* La recuperación ante fallos de ejecución en tareas empleando múltiples modelos de mezcla de gaussianas parametrizados en la tarea, y *iii)* El aprendizaje incremental de trayectorias novedosas en modelos de mezcla de gaussianas parametrizados en la tarea. El funcionamiento de las técnicas propuestas fue probado a través de simulaciones y experimentos con robots reales. La máquina de aprendizaje extremo, aunque requiere un buen número de datos para estimar la cinemática directa, presenta un error bajo cuando se compara con el obtenido por transformaciones homogeneas. Para las propuestas de recuperación a fallos y aprendizaje incremental, se evaluó la tarea de colocar una manga a un maniquí con un manipulador robótico. En la técnica de recuperación de fallos se encontró que la técnica propuesta mejora la realización de la tarea en la mayoría de los casos; y en el aprendizaje incremental, el nuevo modelo parametrizado obtenido después del incremento, presenta mejores respuestas que las logradas empleando el modelo existente.

Palabras clave: Programación por demostración, Máquina de aprendizaje extremo, Recuperación a fallos en robótica, Aprendizaje incremental en robótica.

Contributions in the Generalization of Skills in Robot Learning by Imitation

Abstract

In robot programming by demonstration (PbD) object positions changes related to a task, requires new trajectories that respond to these. One of the existent technique is the task parametrized Gaussian mixture model. This technique allows to relate the robot movements with goals and objects poses, which are called task parameters. One problem that emerge is the need of generalization in cartesian and joint space, specifically it is required to have the direct kinematic model of the robot, with which it is possible to estimate the joints trajectories from cartesian ones. A second problem that arises is that when manipulate deformable object, it is possible to have execution fails, it requires the execution of one of more actions to recovery the fail. A third problem is that, although the generalization technique responds to changes, in certain occasions, is necessary to include new behaviors, which can be different from those already learned. This work focuses on three contributions related to trajectory generalization issue: *i)* The learning and application of the direct kinematics, in PbD using a neural network called extreme learning machine; *ii)* The recovery of execution fails, in tasks programming with multiple task parametrized gaussian mixture models, and *iii)* The incremental learning of novelty trajectory, in task parametrized gaussian mixture models. The proposed techniques were tested through simulations and experiments with real robots. Although the extreme learning machine requires a big number of data to estimate the kinematics, it has a low error, when comparing it with the obtained from homogeneus transforms. For the proposed techniques in fail recovery and incremental learning, the task of putting a sleeve to a mannequin with a robotic manipulator was evaluated. In fail recovery, was found that the technique improving the task performance in most cases; and in the incremental learning, the new task parameterized model obtained after the increase, showed better performance than that of the existent model.

Keywords: Humanoids, Programming by demonstration, Imitation learning, Robot execution fail recovery, Robot incremental learning.

Contenido

Lista de Abreviaturas

Abreviatura	Término
ANN	Artificial Neural Networks
BIC	Bayesian information criterion
DTW	Dynamic Time Warping
DMP	Dynamic movement Primitives
ELM	Extreme Learning Machine
EM	Expectation–maximization algorithm
GA	Genetic Algorithm
GMM	Gaussian Mixture Model
GMR	Gaussian Mixture Regression
HMM	Hidden Markov Model
LWR	Locally Weighted Regression
$LWPR$	Locally Weighted Projection Regression
PpD	Programación por Demostración
$PHMM$	Parametric Hidden Markov Model
RBF	Radial Basis Function
RMS	Root Mean Square
PCA	Principal Component Analysis
$TPGMM$	Task Parametrized Hidden Markov Model

1 Introducción

La Programación por Demostración (PpD) es una técnica que permite a un robot aprender la realización de una tarea a través de la demostración de la misma, por parte de un humano [14]. Se puede aplicar en entornos poco estructurados porque permite que un usuario enseñe fácilmente nuevas tareas a un robot, sin necesidad de conocimientos en programación. En PpD el robot es entrenado con ejemplos de la tarea, a partir de los cuales el aprende a generalizarla.

Desde el punto de vista general, la programación de robots se puede dividir en manual y automática [12], en la Figura **1-1**, se pueden apreciar las tres vertientes de la programación automática. Esta permite que los programas se generen como resultado de la interacción entre un robot y un humano [12]. Los sistemas de aprendizaje crean un programa por inferencia inductiva a partir de ejemplos por parte del usuario, y por exploración por parte del robot. En los sistemas instructivos el programa se genera a partir de comandos de voz o gestos captados visualmente de tareas, estas tareas son previamente programadas con otras técnicas.

Los primeros trabajos en programación por demostración o a veces llamada aprendizaje por imitación, tienen sus comienzos en los años 80, como lo expresaran Schaal [117] y Billard [14], y se dan como una posible solución o ruta alterna que evita la tediosa tarea de programar el robot para tareas complejas. La programación por demostración fue primeramente estudiada en psicología, donde relacionan los comportamientos de aprendizaje con áreas del cerebro, por ejemplo en Schaal [117] y Oztop [103], muestran un modelo del cerebro de un mico, donde se relacionan partes del cerebro con las capacidades visuales y motrices. También, desde el punto de vista

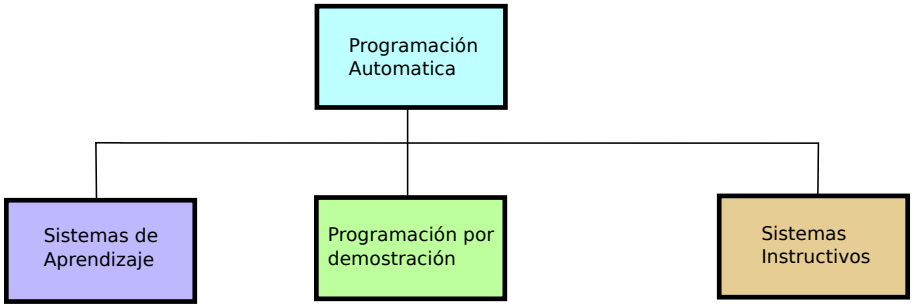


Figura 1-1: Vertientes de la programación automática (Basada en [12]).

Tabla **1-1**: Clasificaciones de la PpD, según Dillman y otros [41].

Complejidad de la tarea	Tareas elementales	Tareas complejas

Tipo de demostración	Activa	Pasiva	Implícita

Representación interna	Trayectorias	Efectos en el ambiente	Operaciones y posiciones de objetos

Estrategia de Mapeado	Directa	Planeada

cognitivo, Brezeal y Scassellati en 2002 [17], presentan una clasificación del aprendizaje social en: imitación, emulación de metas, mejora por estímulo, apoyo social, exposición en ambiente, facilitación social, etc.

En la Tabla **1-1**, se muestran las formas de clasificación de la PpD presentadas por Dillman y otros en 1999 [41], se dividen en: complejidad de la tarea, tipo de demostración, representación interna y estrategia de mapeado. Según la complejidad de la tarea se dividen en tareas elementales como movimientos simples del brazo, o en tareas complejas como por ejemplo el ensamblado. De acuerdo al tipo de demostración: Activa, cuando es realizada directamente por el humano moviendo sus extremidades y captada por sensores o visión; Pasiva, si la demostración se realiza usando un periférico de computador como por ejemplo un ratón; o Implícita, cuando solo se definen las metas que debe cumplir el robot a través de un conjunto de comandos. La representación interna se divide en: Trayectorias, cuando solo se observan y aprenden los movimientos o trayectorias de la mano o dedos; Efectos en el ambiente, cuando se utilizan funciones cognitivas de alto nivel; Operaciones y posiciones de objetos, cuando las demostraciones son transformadas en una secuencia de acciones predefinidas. En cuanto a la estrategia de mapeado, puede ser directa cuando se usa una transformación que mapea movimientos humanos a equivalentes del robot, o planeada cuando se tiene en cuenta las metas y posiciones de los objetos.

Otro punto interesante para aclarar es la diferencia entre el aprendizaje convencional de una función, por ejemplo usando redes neuronales, y el que se realiza en programación por demostración. En el primero se cuenta con los datos (x, y), con $y = f(x)$, x las entradas y y las salidas y f una función, mientras que en el segundo se cuenta con los datos resultado de funciones no lineales de las entradas y salidas $(g(x), h(y))$ [85], donde $g(\cdot)$ es la función de adquisición de los movimientos y $h(\cdot)$ es la función motriz (transforma señales de trayectoria y en movimientos del robot).

Figura 1-2: Robot humanoide PR2.

Una de las áreas de investigación en robótica es el desarrollo de máquinas inteligentes que imiten, de alguna manera, al ser humano, temas como la caminata bípeda, robótica médica, o el aprendizaje por imitación, están en permanente evolución [49]. En particular, en las últimas décadas, el desarrollo alcanzado en dos áreas ha permitido el avance en el desarrollo de robots humanoides (Figura **1-2**): la visión de máquina y la inteligencia computacional o aprendizaje de máquina. Ambas, además, apoyadas en el aumento de las capacidades de cómputo, sirven de fundamento a las investigaciones en aprendizaje por imitación. La visión de máquina ha permitido que el robot aprenda de manera natural "viendo" como lo hace un ser humano. Por su parte, el aprendizaje de máquina permite que toda la información que se produce, tanto en los movimientos humanos como la de posiciones de objetos involucrados en las tareas realizadas por este, sea almacenada y utilizada para producir nuevas acciones no demostradas.

Dentro de las técnicas de aprendizaje de máquina se encuentran, entre otras, las redes neuronales y los modelos probabilísticos. Las redes neuronales permiten realizar un mapeado de entradas a salidas de funciones no lineales, por ejemplo la máquina de aprendizaje extremo, propuesta en el 2004 por Huang [61], la cual es una red de una sola capa oculta cuyas neuronas tienen una función de activación no lineal (Ej. función seno), y las neuronas de salida no tiene función de activación, la estimación se realiza resolviendo un sistema lineal en un tiempo de cómputo muy corto.

En cuanto a los modelos probabilísticos, han sido planteadas técnicas que permiten la generalización de trayectorias, como el modelo de mezcla de gaussianas parametrizado en la tarea. Este

modelo permite relacionar los movimientos del robot con metas y poses de objetos, las cuales pueden ser definidas a través de marcos de referencia. Concretamente, el movimiento del robot es condicionado por un conjunto de variables de la tarea que representan el sistema coordenado de marcos de referencia relevantes. A cambio de representar cada trayectoria como un modelo diferente, el modelo de mezcla de gaussianas parametrizado en la tarea, se basa en un solo modelo que abarca las diferentes trayectorias como función de las variables o parámetros de la tarea. La técnica de modelado, se fundamenta en las propiedades del producto de gaussianas [116].

En muchas ocasiones no es suficiente generalizar trayectorias, también es necesario que el robot pueda modificar su comportamiento ante cambios externos, como por ejemplo recuperarse ante fallos de ejecución o aprender nuevos comportamientos. En el primer caso, la recuperación a fallos de ejecución, permite que ante un fallo o problema el programa del robot busque solucionarlo; por ejemplo para la tarea de colocar la manga de una camisa a un maniquí por un robot, si esta manga se engancha, el robot regresa por la misma trayectoria de colocación y la repite tratando de corregir el problema. Por otro lado, el aprendizaje incremental, permite que se agregue nueva información al modelo de la tarea; retomando el ejemplo de la manga de la camisa, si el robot aprendió a colocarla con el brazo orientado hacia adelante, este permite que además aprenda a colocarla con el brazo orientado hacia atrás.

1.1. Objetivos

1.1.1. General

Diseñar técnicas que permitan la recuperación a fallos de ejecución y aprendizaje incremental de una habilidad en Aprendizaje por Imitación de robots.

1.1.2. Específicos

- Implementar al menos un esquema de aprendizaje por imitación.
- Diseñar una técnica para la recuperación a fallos de ejecución usando múltiples modelos parametrizados en la tarea.
- Diseñar técnicas que permitan incrementar trayectorias a un modelo de mezclas parametrizado en la tarea.
- Definir una métrica que permita evaluar resultados en aprendizaje por imitación.

1.2. Contribución

Este trabajo se centra en tres aportes realizados en programación por demostración: *i)* Aprendizaje del modelo cinemático usando una máquina de aprendizaje extremo; *ii)* Recuperación ante fallos de ejecución en tareas que involucran la generalización de trayectorias; y *iii)* Incremento de nuevas trayectorias en técnicas que permiten la generalización de trayectorias.

Aprendizaje del modelo cinemático usando una máquina de aprendizaje extremo en programación por demostración
El aporte consiste en el empleo de la máquina de aprendizaje extremo para aprender el modelo cinemático directo de un robot manipulador de cinco grados de libertad. A través de un programa, el robot reproduce trayectorias de articulación y se obtienen las trayectorias de las posiciones del efector final del robot, con esta información la red neuronal aprende la función no lineal entre la entrada (articulaciones) y la salida (posición del efector final). Este modelo cinemático es luego usado para generalizar el punto final de una trayectoria.

Recuperación ante fallos de ejecución
En ciertas tareas, como la tarea de poner una manga de camisa con un brazo robótico, se presentan una serie de retos interesantes. Por un lado, el robot debe operar cuidadosamente cerca de un humano cuyo brazo puede estar en posiciones diferentes. Por otro lado, cuando se pone la manga de una camisa, frecuentemente se engancha la camisa en el codo. Una persona, cuando pone la camisa, detecta el enganche y de forma natural retrocede, modifica un poco la posición, y retoma la trayectoria cambiándola ligeramente. En este caso, el ser humano, realizó la recuperación del fallo de ejecución (enganche), al retroceder e intentar una nueva trayectoria. El aporte consiste en el empleo de múltiples modelos de mezcla de gaussianas parametrizados en la tarea, para la recuperación ante un fallo de ejecución. Un árbol de búsqueda es estimado para determinar cual modelo de mezcla parametrizado usar según el fallo que se presenta. Además, el sistema es ampliable a nuevos fallos.

Incremento de trayectorias novedosas en técnicas que permiten generalizar trayectorias
El aprendizaje incremental permite que se agregue nueva información de la tarea a un modelo. La contribución consiste en tres técnicas: generativa, directa y adición de modelos, que permiten obtener nuevos modelos de mezcla de gaussianas parametrizados en la tarea. Esto facilita el aprendizaje incremental y el re-entrenamiento por demostración del robot, ya que no es necesario enseñarle nuevamente todas las trayectorias, o tenerlas almacenadas para su estimación. Aunque las técnicas generativa y de adición de modelos sí requieren almacenar unos pocos valores, como los parámetros de la tarea de trayectorias previas, la técnica de actualización directa no lo requiere.

1.3. Organización

En el Capítulo 2, se presentan las bases de la PpD y los temas relacionados con las contribuciones de la tesis. En el Capítulo 3 se presenta la teoría de aprendizaje incremental de modelos de mezcla de gaussianas y de los modelos de mezcla de gaussianas parametrizados en la tarea, así como la máquina de aprendizaje extremo. El uso de la máquina de aprendizaje extremo, que permite generalización de una trayectoria a partir de la obtención del modelo cinemático directo con la red neuronal, es presentado en el Capítulo 4. En el Capítulo 5, se trata lo referente a la recuperación ante fallos de ejecución y se prueba con la tarea de colocar una manga de camisa en un maniquí. En el Capítulo 6, se exponen las técnicas que permiten incrementar trayectorias novedosas a un modelo de mezcla de gaussianas parametrizado en la tarea. Por último, se presentan las conclusiones.

2 Programación por Demostración de Robots

La PpD de un robot puede variar según la complejidad de la tarea. Si se trata de tareas elementales, generalmente consiste en la adquisición, adecuación y modelado de las demostraciones, usando máquinas de aprendizaje y luego se realiza la reproducción. En el caso de tareas complejas, el procedimiento consiste en segmentar las trayectorias en porciones o acciones básicas, luego generar una representación simbólica de la tarea y por último realizar su reproducción. Una variación a lo anterior es el aprendizaje basado en metas, donde no se aprenden las trayectorias de la tarea sino las metas de la tarea que el robot debe realizar.

En la Sección 2.1 se presentan temas relacionados con la programación por demostración, y se ilustran los componentes del proceso: Primero se realiza la adquisición y adecuación de la información de las demostraciones; Luego el modelado de la información, que en el caso de una tarea compleja requiere la segmentación y su representación simbólica; Por último, el robot realiza la reproducción de estos modelos o listado de símbolos. Y para cerrar, en la Sección 2.2 se presentan los trabajos relacionados con el aporte de la tesis, como: generación de nuevas trayectorias a partir de las demostradas, tareas complejas, recuperación ante fallos y aprendizaje incremental.

2.1. Bases de la Programación por Demostración

Los dos diagramas de bloque de la Figura **2-1**, representan los componentes de la PpD según la complejidad de la tarea. La adquisición de la información de las demostraciones, se realiza tanto para tareas elementales como complejas. Para tareas elementales, estas demostraciones son codificadas y/o modeladas usando técnicas de aprendizaje como redes neuronales o modelos ocultos de Markov, con los cuales es posible generalizar las trayectorias y reproducirlas en el robot. En el caso de tareas complejas, es necesario dividir o segmentar la tarea en subtareas, para luego generar una representación simbólica de la secuencia de la tarea, que se reproducirá por el robot.

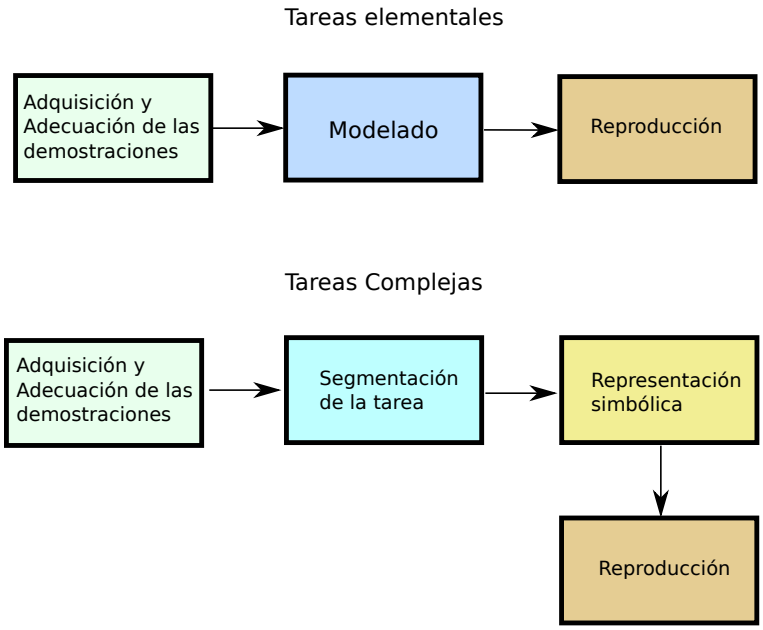


Figura 2-1: Diagramas de bloques que componen la PpD, según la complejidad de la tarea.

2.1.1. Adquisición de la Información

La adquisición de la información de las demostraciones, es el primer paso en PpD, esta información proviene de sensores y permite, luego en la etapa de reproducción, el modelado de los movimientos humanos. Desde el punto de vista de la captura de la información, se pueden dividir en:

- Sensado directo sobre el robot.
- Sensado directo sobre el humano.
- Sensado indirecto de los movimientos humanos.

Sensado Directo Sobre el Robot
El humano guía directamente componentes del robot, para esto el robot debe tener articulaciones no rígidas o de baja relación de engranaje. En las demostraciones cinestáticas como en Billard [25], Lee [78], Nemec y otros [99], el efector final del robot es guiado por un humano y los valores angulares captados por los codificadores rotatorios de las articulaciones del robot son almacenados. En la Figura **2-2**, se muestra un ejemplo de entrenamiento cinestático. Otra forma, es el uso de sensores de fuerza/torque [109], para guiado del efector final o también con interfaces táctiles (piel sintética), las cuales han sido presentadas en [104], las trayectorias son captadas según la fuerza y posición de la mano humana en la piel sintética colocada en el robot.

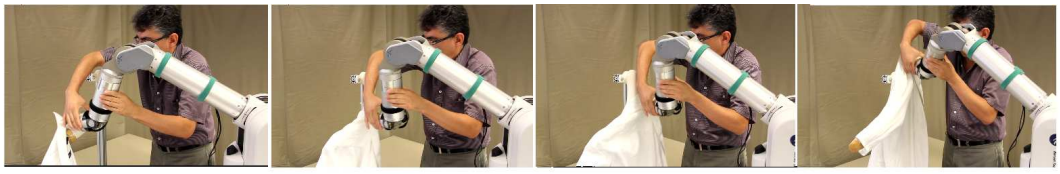


Figura 2-2: Demostración cinestática con el robot WAM.

La ventaja del sensado directo cinestático, es que no requiere resolver el problema de correspondencia y, la desventaja, es que puede ser difícil que una persona mueva simultáneamente varias extremidades o incluso solo la mano al mismo tiempo. En cuanto las técnicas por sensores de fuerza y táctiles su desventaja esta en que las trayectorias se registran sólo en el plano cartesiano y no en el de articulaciones.

Sensado Directo Sobre el Humano
A través de sensores colocados de manera directa sobre el humano, se obtienen los movimientos. Por ejemplo en [122], se utilizan sensores magnéticos de movimiento, en [69] utilizan unidades de medida de inercia, también para capturar el movimiento de los dedos, agregan a lo anterior un guante de datos como en: [122], [69], [73]. Otra forma es el empleo de comandos de voz como en [130]. Por último, a través de estructuras articuladas con sensado por codificadores rotatorios, captan los movimientos del humano por tele-operación como en [111] y [116].

La ventaja de este tipo de sistemas es la alta velocidad con que se pueden registrar los movimientos, como desventajas es su costo, el tiempo que se requiere para ubicarlos en el cuerpo y que con algunos es necesario resolver los problemas de correspondencia.

Sensado Indirecto de los Movimientos Humanos
Usando visión de máquina se realiza la adquisición visual de los movimientos del demostrador. Con una sola cámara, se han utilizado modelos cinemáticos de la mano [135], o filtro de partículas para capturar los movimientos de las extremidades [9], o flujo óptico [76] para la obtención de movimientos de la mano. Sistemas de visión con varias cámaras, donde se utiliza un sistema de 12 cámaras (OptiTrack) para capturar las posiciones de objetos [116]. A partir de la aparición del sensor activo Kinect, se han expuesto trabajos donde lo emplean, como [39], [82], [127], en estos capturan las trayectorias de objetos que son desplazados por una persona, o las articulaciones de una persona [138], [126], o la mano y los objetos [77]. También han sido usadas cámaras de tiempo de vuelo, las cuales reconstruyen imágenes tridimensionales de objetos o personas [108].

La ventaja de este tipo de sistemas es que captan la información directamente del demostrador de manera similar a como lo hacen los seres humanos, como desventajas es que pueden ser ocluidos, algunos tienen un alto costo y requieren ubicarse de forma específica (Ej. Optitrack), en otros casos

su baja velocidad (cámaras simples, par estéreo, Kinect). Por último, en la mayoría de los casos, es necesario resolver el problema de correspondencia de las articulaciones del humano con las del robot.

2.1.2. Adecuación de la Información

Es el procesamiento que se debe realizar a las trayectorias demostradas por ser humano para eliminar el ruido de estas. Esta adecuación permite obtener mejores modelos de las demostraciones. Asada y Liu [8] plantean que para el entrenamiento de una red neuronal, los datos deben cumplir la condición de continuidad de Lipschitz, de lo contrario es necesario eliminar los datos erráticos (filtro). También Kaiser y Dillman [68], hablan sobre el preprocesamiento que se debe realizar a las señales tomadas de un ser humano, ya que estas contienen perturbaciones o incluso acciones incorrectas y acciones tipo escalón. Los autores proponen, identificar las percepciones relevantes $x(t)$ que producen cambios significativos en las acciones $u(t)$, además verificar los datos con la condición de Lipschitz, para eliminar datos erráticos:

$$||x'(t_1) - x'(t_2)|| \leq \delta||u'(t_1) - u'(t_2)||, \tag{2-1}$$

donde $\{x'(t), u'(t)\}$ es el conjunto de datos identificados y filtrados para realizar el entrenamiento.

Problema de la Correspondencia
Aparece ante las diferencias cinemáticas entre las articulaciones humanas y las de un robot, por ejemplo las dimensiones físicas o la cantidad de grados de libertad del cuerpo humano con respecto a los grados del robot [98] y [3]. Su solución, permite identificar que secuencia de acciones del imitador es más cercana a la demostrada [5].

Argall [6] divide la correspondencia en dos tipos de mapeo, mapeo de registro y mapeo de personificación. El mapeo de registro se refiere al registro exacto de las acciones demostradas por el maestro. El mapeo de personificación trata de las acciones que el observador, a partir de la demostración, adapta a su cuerpo. Por esto el autor diferencia demostración e imitación, demostración es cuando solo hay mapeo de registro, lo que significa equivalencia física del humanoide con el ser humano, e imitación cuando hay un mapeo de personificación.

Otro problema interesante, que guarda alguna similitud con la correspondencia, son las diferencias dinámicas de las extremidades humanas con las de los robots actuales, por ejemplo hay variaciones de parámetros como longitud, peso, velocidad de respuesta y capacidad de levante de peso [79], [95].

2.1.3. Modelado de la Información

Un modelo cinemático del cuerpo humano puede tener muchos grados de libertad [42], si se tiene en cuenta que cada uno de ellos genera una trayectoria, existe una gran cantidad de información que debe ser aprendida por una máquina. Las técnicas de modelado permiten codificar toda esa información en modelos compuestos por pocos valores. En PpD se han trabajado diversas técnicas, pero las más usadas son: redes neuronales, aprendizaje probabilístico (modelos ocultos de Markov, modelos de mezcla de gaussianas, redes bayesianas), primitivas dinámicas de movimiento, y mezcla de alguno de los anteriores con aprendizaje por refuerzo. Esta parte del proceso de PpD es la que se encuentra más relacionada con esta tesis, ya que en ella se trabajó el modelado con una red neuronal y el incremento de trayectorias novedosas en modelos de mezcla de gaussianas parametrizados en la tarea.

Redes Neuronales
Las redes neuronales permiten aprender mapas de entradas a salidas de funciones no lineales. Son sistemas compuestos por neuronas que se interconectan para producir un estímulo de una o más salidas. Trabajos en PpD usando diversas redes neuronales, son por ejemplo: feedforward [8], de base radial [68], recurrentes [66] y [64], Hopfield y similares [31] y [76], de regresión generalizada [102] y [128].

Kaiser y Dillman en 1996 [68], presentaron una red neuronal de funciones de base radial que aprende por demostración tareas como: insertar una pieza en un hueco y abrir la puerta de un gabinete pequeño usando un manipulador robótico. La entrada de la red son las percepciones y, y la salida las acciones u. Antes del entrenamiento, se identifican las variables relevantes y se filtran, luego, algunos parámetros como los centros y anchos de las funciones de la red se calculan con un algoritmo de agrupamiento "clustering", y los pesos con gradiente descendente. Sus principales ventajas son que puede aprender incrementalmente y se adapta a variaciones en la meta de la tarea y como desventaja es el tiempo que tarda en su entrenamiento.

Las aplicaciones en PpD con redes neuronales recurrentes tienen la ventaja de utilizar menos neuronas comparadas con las que utilizan redes tipo Hopfield y similares. Por ejemplo, en el trabajo presentado por Ito y Tani [66], emplean alrededor de 100 neuronas para programar un pequeño robot humanoide QRIO de SONY, que aprende cuatro movimientos de ambos brazos. Debido a que requieren aprendizajes iterativos, su desventaja es el tiempo que tardan en su entrenamiento.

El uso de imágenes como entrada a un red neuronal, ha sido presentado en Chaminade y otros [31] donde binarizan una imagen de 32×12 bits, y modelan los movimientos de los dedos de una mano. Su desventaja esta en que trabajar con los píxeles de las imágenes directamente, requiere un gran poder de cómputo.

El flujo óptico para entrenar una red neuronal ha sido usado en [76] y [102]. En 2003 Kuniyoshi y otros [76], emplearon una versión de la red Hopfield llamada “non-monotonic neural net", la cual aprende gestos con la mano. Como ventaja es su robustez al ruido de las trayectorias demostradas y sus desventajas es que presenta problemas ante estados estáticos de las trayectorias y debido a que requiere múltiples neuronas (1000 en total), es necesario un gran tiempo de entrenamiento. En 2011 Nope y otros [102], usan una red de regresión generalizada en PpD, a través de cuatro gestos realizados con las manos, evalúa su sistema tanto de manera cualitativa (encuestas a personas), como cuantitativa comparando el gesto con modelos geométricos, y concluye que ésta los aprende. Como ventaja es que la red neuronal resuelve el problema de correspondencia.

Arquitecturas compuestas por múltiples redes neuronales en PpD han sido presentadas en [44] y [113]. En 2005 Erlhagen y otros [44], presentaron una técnica que usando redes de campo dinámico (Dynamic Field) con aprendizaje Hebbiano, aprenden la meta de una tarea, usando información visual del objeto. La meta consiste en tomar el objeto y llevarlo hasta una plataforma elevada, según el color del objeto, hay dos tipos de agarre y dos plataformas. Su ventaja consiste en que puede trabajar con información visual imprecisa. En 2013 Rempis [113] presentaron una arquitectura compuesta por múltiples redes neuronales. Cada red neuronal es de anillo de memoria (NRM del inglés Neural Ring Memory), y cumple funciones específicas como: captura movimiento, memorización, reproducción de movimiento. La aplicaron a movimientos simples del brazo de un robot humanoide. Una de las ventajas de esta técnica, según los autores, es que puede almacenar múltiples movimientos usando la misma arquitectura.

Modelos Ocultos de Markov

Abreviado como HMM (del inglés Hidden Markov Model), es un método probabilístico que en PpD permite codificar las variaciones temporales de movimientos humanos a partir de varias demostraciones de los mismos [65]. Se han usando dos versiones: discretos [65], [18] y continuos [20]. En PpD ha sido utilizado el tipo de modelo izquierda-derecha, en el cual las interconexiones de un estado a otro pasan de izquierda a derecha (Figura **2-3**). En el caso discreto, el sistema se compone de estados, conectados por probabilidades $a_{i,j}$ de transición entre ellos, a cada estado se le asocia un vector de salidas b_j. En el caso de la representación izquierda derecha, el cambio de estado en el tiempo, genera como salida cada uno de los valores, formando las muestras de la trayectoria. En el caso continuo, ante cambio de los valores de salida se tienen un vector de medias μ_j y una matriz de covarianza $\Sigma_{i,j}$ de gaussianas asociadas a cada estado, con j el estado actual e i el estado anterior (Figura **2-3**).

A partir de dividir el movimiento de una articulación en tramos y usar un estado del modelo HMM discreto para representar una muestra clave del tramo, se han desarrollado trabajos como el de Inamura [65] o los de Calinon [18] y [26].

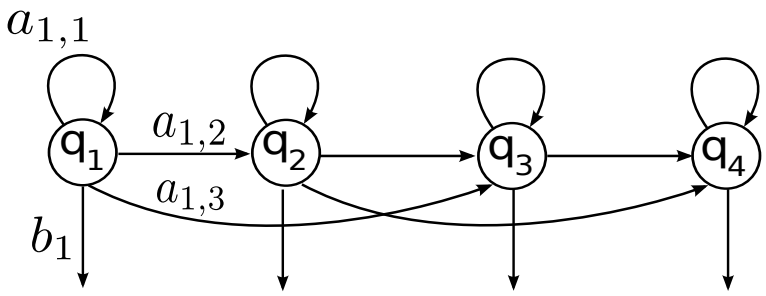


Figura 2-3: Modelo HMM izquierda-derecha.

Una comparación de modelos ocultos de Markov combinado con regresión gaussiana (GMR) contra otras técnicas fue presentada por Calinon en 2010 [20], las técnicas que compara son regresión local ponderada (LWR), "locally weighted projection regression" (LWPR), regresión gaussiana dependiente del tiempo (TGMR), primitivas dinámicas de movimiento (DMP). El autor muestra mediante resultados de cinco métricas que la combinación es ventajosa.

La ventaja de HMM continua, es que es capaz de representar las variabilidades de múltiples demostraciones de la misma tarea y varias trayectorias de articulación [20], y como desventaja, es que requiere un tiempo grande para su estimación.

Modelo de Mezcla de Gaussianas

Abreviado como GMM (del inglés Gaussian Mixture Models) [15], es un modelo probabilístico similar a HMM continuo, pero donde solo se conectan los estados adyacentes, éste permite codificar una o más trayectorias a un modelo compuesto por K gaussianas (estados), cada una de ellas con parámetros: α_i coeficiente de mezcla, $\boldsymbol{\nu}_i$ vector de centros y $\boldsymbol{C}_i$ matriz de covarianza de la *i-ésima* gaussiana. Los valores iniciales de los parámetros se obtienen, generalmente, usando el algoritmo de *k-medias* de los datos $\boldsymbol{\xi}$, también se han usado valores iniciales obtenidos al realizar particiones iguales a la cantidad de gaussianas K, en el tiempo de los datos [24]. Si se define $\boldsymbol{\Theta}$ como los parámetros del modelo:

$$\boldsymbol{\Theta} = \{\alpha, \boldsymbol{\nu}, \boldsymbol{C}\}. \tag{2-2}$$

$$\boldsymbol{\Theta_0} = kmedias(\boldsymbol{\xi}, K). \tag{2-3}$$

Luego estos valores son refinados usando el algoritmo EM (Esperanza-Maximización) [24], el cual se compone de dos pasos. En el primer paso (E), se halla el valor de probabilidad suponiendo que se conocen los parámetros del modelo. En el segundo paso (M), se buscan parámetros a partir de ecuaciones obtenidas al derivar el logaritmo de la probabilidad (maximización).

$$\boldsymbol{\Theta} = EM(\boldsymbol{\xi}, \boldsymbol{\Theta_0}). \quad (2\text{-}4)$$

En 2007 Calinon y otros [27], utilizaron las GMM para extraer las características importantes de varias demostraciones de una misma tarea. La información de las trayectorias de la tarea, son reducidas en dimensionalidad con PCA (del inglés Principal Component Analysis) y luego codificadas con GMM.

La ventaja es que es capaz de representar las variabilidades de varias demostraciones de la misma tarea y varias trayectorias de articulación y como desventajas, es que requiere un tiempo grande para su estimación y debe incluir en el modelo la variable tiempo, ya que por si solo no es capaz de modelar variaciones temporales.

Red Bayesiana y Aprendizaje Basado en Metas

Las redes bayesianas son una representación gráfica de dependencias de información, en la cual los nodos representan variables y los arcos representan relaciones de dependencia directa entre las variables [86]. Una red bayesiana, queda definida por su estructura y las probabilidades condicionales entre un nodo y sus padres. Son utilizadas en razonamiento probabilístico para la solución de problemas complejos, como inferir que hacer ante ciertos valores tomados por las variables. En la Figura **2-4**, se muestra una red bayesiana para la usabilidad de un objeto [86]. Esta red una vez entrenada, puede inferir a partir de datos de entrada como: características de un objeto (Ej. Forma, Tamaño) y las acciones (Ej. agarrar), que efectuar sobre el objeto (Ej. contacto).

Usadas generalmente para el aprendizaje basado en metas, en esta técnica, el modelo es entrenado para realizar acciones y lograr una meta. En 2004 Bentivegna y otros [10] proponen un sistema basado en primitivas y un generador de acciones. El robot aprende a jugar hockey de mesa, a partir de demostraciones obtienen las acciones primitivas que el robot debe realizar ante ciertos eventos. Luego una técnica de regresión local es usada para generar la acción a realizar a partir del evento.

Grimes y otros [51], usan redes bayesianas no-paramétricas para enseñar a un robot a balancearse en una pierna y a ponerse en cuclillas, la red es estimada con regresión de procesos gaussianos. En Montesano y otros [86] plantean redes bayesianas para la potencialidad o usabilidad de objetos, la estructura de la red es ajustada usando cadenas de Markov montecarlo y el algoritmo K2 [36], el cual encuentra la mejor estructura a partir de una base de datos de casos del problema.

Las redes bayesianas son empleadas por Kragic y otros [72], para modelar la tarea de agarrar objetos, o por Shon y otros [121], para tomar y levantar una cubeta con los dos brazos, para armar figuras de dos dimensiones a partir de múltiples demostradores humanos en internet Jae-Yoon y otros [34].

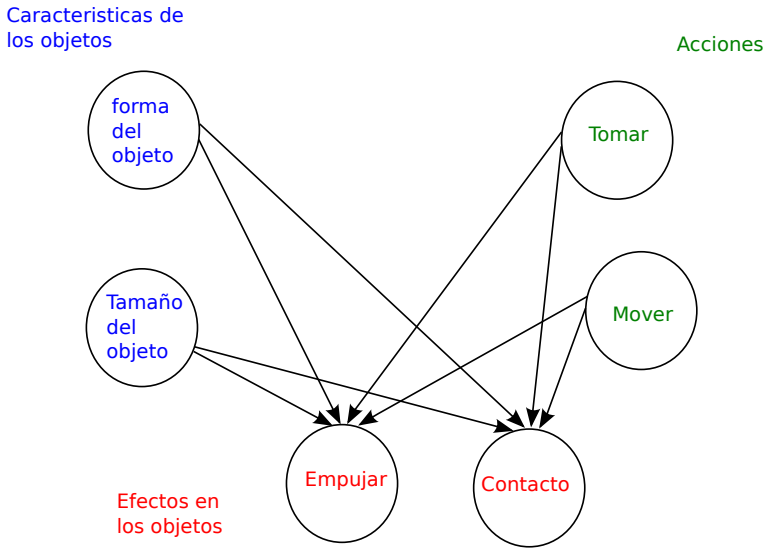


Figura 2-4: Red bayesiana para la usabilidad de un objeto [86]. En azul, se muestran las características de los objetos; en rojo, los efectos en los objetos; en verde, las acciones.

Como ventajas de las redes bayesianas, es su fácil comprensión y las variables pueden ser entradas o salidas según el problema que se tenga, como desventaja es que para muchas variables se puede volver compleja la estimación de la estructura, como en [125].

Primitivas Dinámicas de Movimiento

Las primitivas dinámicas de movimiento abreviadas como DMP (del inglés Dynamic Movement Primitives) [63], [118] y [119], se forman a partir de una ecuación diferencial a la que se le suma una función no lineal $f(x)$, la cual a su vez es la suma ponderada de funciones no lineales y x es definida por $\tau\dot{x} = \alpha_x x$, con α_x una constante.

$$\tau\dot{z} = \alpha_z(\beta_z(g - y) - z) + f(x), \tag{2-5}$$

$$\tau\dot{y} = z \tag{2-6}$$

donde z es una variable base que permite generar la trayectoria de salida deseada y, las constantes τ, α_z, β_z, modifican el comportamiento de la salida, y g es el valor final deseado de esta. La función $f(x)$, es la suma de funciones ponderadas por el valor de w, resultando en una señal compleja que se comporta como la trayectoria.

El ajuste de los valores w, se hace usando regresión por mínimos cuadrados incremental “Incremental Least Squares Regression” [63] o regresión local ponderada “Locally Weighted Regres-

sion" [119] con la información de la trayectoria original. Como precisa Ijspeert [63] el sistema dinámico codifica trayectorias deseadas de las articulaciones, no señales comando para los motores de las articulaciones, además su gran ventaja es la robustez del modelo a señales ruidosas, la cual es demostrada en un análisis de estabilidad.

Sus ventajas es que es fácil y rápido de estimar, no depende de manera directa del tiempo, lo que le permite que se pueda detener la ejecución del movimiento. Como desventajas está en que se requiere un modelo por cada trayectoria de articulación o cartesiana y que por ser una técnica que representa los movimientos de manera determinista, no es capaz de representar variabilidades del movimiento [35]. Esto último ha sido corregido con técnicas como DMP correlacionado [22], donde la variabilidad y correlación de las diferentes demostraciones a lo largo del movimiento es tomada en cuenta en el modelo, también en la propuesta del 2014, llamada DMP probabilístico [35].

2.1.4. Segmentación y Representación Simbólica de la Tarea

La segmentación de la tarea permite descomponer la demostración de una tarea compleja en subtareas. Tomando como ejemplo la tarea de apretar una tuerca con una llave usando un robot humanoide, las subtareas podrían ser: mover la mano del robot a un punto cercano a la llave, tomar la llave, mover la mano hasta un punto cercano a la tuerca, casar la llave en la tuerca, girar la llave para apretar. Cada subtarea tiene una denominación o etiqueta, por ejemplo tomar la llave. Aunque en esta tesis no se emplea la segmentación de tareas y la representación simbólica, estas se tratan a continuación.

Segmentación de la Tarea
En 2002 Fod y otros [47], presentaron una técnica que deriva primitivas de la tarea a partir de un movimiento completo, primero segmentan la tarea usando la suma al cuadrado de las velocidades de las articulaciones que describen el movimiento. Dadas m articulaciones $\boldsymbol{q}$, donde cada articulación tiene N muestras, la suma del cuadrado de las velocidades de articulación es:

$$v_s(k) = \sum_{i=1}^{m} \dot{\boldsymbol{q}}_i{}^2(k), \qquad k = 1, ..., N. \tag{2-7}$$

Segundo, a partir de detectar cuando la suma al cuadrado de las velocidades es menor a un umbral, separan los segmentos o subtareas. En la Figura **2-5** se muestra un ejemplo de las trayectorias de cuatro articulaciones y en la parte inferior en color azul, la suma al cuadrado de velocidades obtenida y la señal umbralizada en color rojo. El valor de umbral es obtenido a partir de múltiples demostraciones repetidas y ajustado para identificar la mayor cantidad de segmentos para todas las demostraciones repetidas. Luego, usando análisis de componentes principales, encuentran las

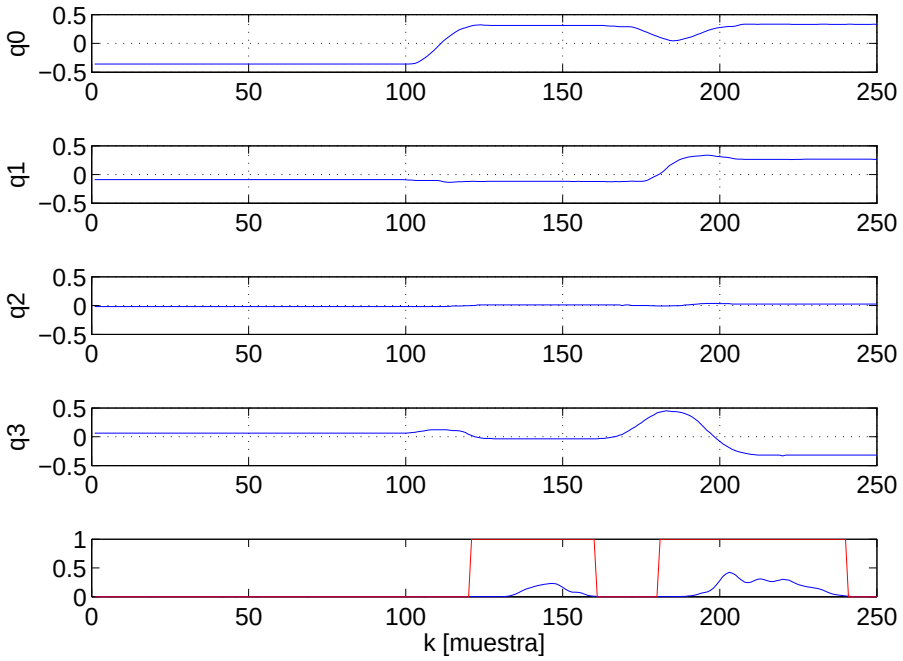


Figura 2-5: Trayectorias de articulación de una demostración. Ejes q_0 a q_3 corresponden a las cuatro articulaciones del brazo. En la parte inferior, se muestra la suma de velocidades de las articulaciones (azul) y en rojo la señal de pulsos resultado de aplicar el umbral.

primitivas del movimiento completo. Su ventaja es la simplicidad, su desventaja es que por depender de un umbral, el sistema solo alcanza un 80 o 85 por ciento en la identificación de las subtareas.

Kadone y otros en 2006 [67], emplearon una red neuronal del tipo memoria asociativa, para etiquetar segmentos de una sucesión de subtareas. Segmentan los tramos de la tarea usando dos ventanas que extraen porciones de la sucesión, estos tramos son comparados por correlación y almacenados por la memoria asociativa. El sistema permite que nuevas subtareas sean aprendidas de manera incremental por la memoria asociativa. La desventaja de la técnica anterior es que en el entrenamiento inicial, un tramo de movimiento o subtarea debe repetirse en más de una vez en la sucesión de tareas, para poder ser aprendido por la memoria asociativa.

En 2011 Meier y otros [90], utilizaron una librería de acciones o subtareas modeladas con DMP y una técnica donde reformulan la técnica de DMP, para que reciba como entradas la subtarea y la librería de modelos y retorne la subtarea más probable. Realizan pruebas con la escritura de palabras cursivas y la tarea de servir líquido usando un robot WAM [132]. Su ventaja reportada es que la rata de reconocimiento llega a un $90\,\%$, y su desventaja es que requiere una librería de acciones previo al proceso de segmentación.

Kulic y otros en 2012 [84], presentaron un algoritmo que segmenta tareas a partir de combinar segmentación por cruce por cero de la velocidad (ZVC del inglés Zero Velocity Cross) de la articulación y modelos ocultos de Markov. Primero identifican los grados de libertad más significativos de la tarea, usando la desviación estándar. Luego caracterizan el punto de segmentación mediante una combinación de ZVC y los picos de velocidad de articulación. En una segunda pasada comparan la similitud de probabilidad de los segmentos obtenidos en el paso anterior, con ejemplares previamente entrenados. En 2013 Kulic y otros [33], propusieron una técnica similar a Mataric [47], en esta segmentan la tarea usando la velocidad de las articulaciones, obteniendo las subtareas, las cuales son clasificadas usando una técnica de "clustering" llamada propagación de afinidades AP (del inglés Affinity propagation), con la que reconocen a qué movimiento pertenece cada segmento, al encontrar el respectivo "cluster". El algoritmo AP es entrenado usando una base de movimientos ejemplares. Aunque los autores reportan mejoras en la identificación de las subtareas con respecto a [47], su desventaja es que requiere una base inicial de movimientos ejemplares.

Representación Simbólica de la Secuencia de la Tarea
Con la segmentación de la tarea, se genera un plan de acciones, con el que es posible generar una secuencia de alto nivel que indica las operaciones a realizar. Para lo anterior, se han usando: gramática propia [2], gramática de movimiento [39], autómatas de estado finito [43], [100], redes de Petri [32]. Este tema, se ampliara más adelante en la Sección 2.1.6, que trata con tareas complejas.

2.1.5. Reproducción de Movimientos

Esta reproducción recrea, a partir del modelo o representación simbólica, la trayectoria que fue codificada a partir de las demostraciones. Generalmente estas reproducciones son comparadas con las demostraciones a través de métricas (tema presentado en la Sección siguiente). Para la reproducción de movimientos en ambientes con humanos, es necesario que el robot cuente con un control "compliance" [38], éste, generalmente, se implementa sobre robots cuyas articulaciones no son rígidas, como el robot WAM [132] o el LWR de KUKA [1]. El controlador se diseña para que reproduzca las trayectorias de articulación, pero teniendo en cuenta las fuerzas en las articulaciones. Su gran ventaja es que ante el contacto de un componente del robot con un ser humano u otro objeto, éste es permisivo en la ejecución de la trayectoria y detiene el movimiento.

2.1.6. Tareas Complejas

Las tareas complejas son aquellas que se pueden descomponer en subtareas, por ejemplo el ensamblado de un carro de juguete en madera. Dadas una serie de piezas bien elaboradas, la meta de un ensamble es juntarlas o acoplarlas para producir un objeto que funcione [137]. El proceso

1	Move_xy(x_i,y_i)
2	Move_z(z_i)
3	AttachObj
4	Move_z(TOP)
5	Move_xy(x_f,y_f)
6	Move_z(TABLE)
7	DetachObj
8	Move_z(TOP)

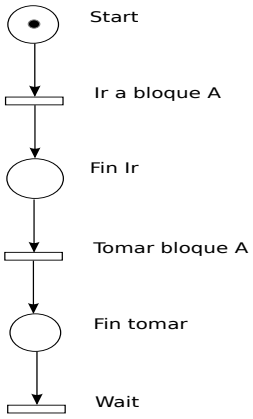


Figura 2-6: Ejemplos de representación simbólica. Izquierda: Gramática, derecha: Red de Petri.

total de ensamble se puede dividir en tareas o sub-ensambles. A partir de la segmentación de la tarea o de extraer la secuencia de acciones, se genera la representación simbólica de la secuencia (Figura **2-6**), se ha usado: gramática como en [2], [39], [43], [88], grafos como en [32], [100], o modelos ocultos de Markov [74].

En 2003 Aleotti y otros [2], emplearon una gramática propia para la tarea de apilar bloques. Las acciones de cada subtarea son escritas manualmente, la secuencia es obtenida desde las demostraciones; debido a que requiere de una librería de operaciones simples escritas manualmente, este trabajo es limitado. Ekvall y Kragic en 2006 [43] también emplean una gramática propia para la tarea de colocar los cubiertos y platos en un puesto de la mesa. La posición y orientación de un objeto conforma un estado, a partir de los estados y varias demostraciones, se generaliza la tarea, que consiste en ordenar elementos comunes y cumplir las restricciones en el orden de ejecución. Su ventaja es la simplicidad y su desventaja es que solo puede haber un objeto por clase. En 2014 Martinez y otros [88], emplearon una gramática propia y aprendizaje por refuerzo para una tarea de ensamble. Con el movimiento y contacto de los objetos a ensamblar construyen unos grafos, con los cuales entrenan un algoritmo de aprendizaje por refuerzo relacional, según las pre-condiciones y efectos deseados más adecuados, el algoritmo decide qué acción ejecutar. Su ventaja está en que es permisivo a variaciones de la percepción de los movimientos, partes no identificadas y errores de inserción.

Dantam y otros en 2012 [39], usaron gramática de movimiento (motion grammar), para un ensamble simple de una estructura de barras de madera. El sistema convierte los movimientos de los objetos capturados usando un sensor kinect, en códigos que luego el robot reproduce, pero no aborda los problemas de forma de agarre y fuerzas que se presentan, además el sistema aprende una sola secuencia de ensamble.

Las redes de Petri son empleadas por [32], para tomar, mover, y apilar bloques de madera. La técnica reconoce los objetos y su ubicación (estado), las variaciones de estos estados y genera nuevos nodos de la red. Su desventaja es que la generación de los nodos depende de los objetos y no se diseñó para aprender manipulaciones que no involucren objetos. Niekum y otros [100] emplean un autómata de estados finitos para la tarea de enroscar la pata a una mesa. A través de aprendizajes interactivos por parte del ser humano, puede aprender de manera incremental a tomar la pata de la mesa de otras ubicaciones y orientaciones, esto implica adicionar ramas al autómata de estados finitos, las cuales luego son simplificadas con las ramas iniciales en sus tramos similares.

En 2006 Kruger y otros [74], presentaron una técnica que detecta primitivas de movimiento de manera no supervisada y con ellas sintetizan tareas. Las primitivas son reconocidas y modeladas usando modelos ocultos de Markov paramétricos (PHMM por sus siglas en inglés), además para la detección de las primitivas, se valen del estado de los objetos. Los autores plantean una dualidad entre el movimiento de los objetos y las acciones realizadas por el humano, basados en esto, usan la segmentación del movimiento de los objetos, para separar las acciones del humano en primitivas. Los autores no ahondan en el tema de la obtención de la gramática que se genera a partir de la tarea.

2.1.7. Métricas en Aprendizaje por Demostración

Existen pocos trabajos dedicados solo al tema de métricas en programación por demostración, esto debido a que cada autor realiza sus pruebas sobre un sistema de adquisición y robot específico, y no sobre un sistema genérico. En Pomplun y Mataric [110], realizan un estudio del movimiento del brazo humano, usando cuatro sensores de movimiento. Realizado con 11 individuos y uno de contraste, plantean una métrica basada en el conjunto de valores que toman los ángulos de las articulaciones, este estudio solo trata el problema de métricas desde el punto de vista de la adquisición de los movimientos.

El problema de correspondencia influye en el cálculo de las métricas como se plantea en [3] y [87]. En 2006 Alissandrakis y otros [3], desarrollan una teoría para calcular métricas cuando se presenta el problema de correspondencia, los autores plantean unas ecuaciones matemáticas que convierten las acciones del demostrador a acciones del imitador. El uso de las métricas se ilustra comparando un humano contra robots diferentes como: el robot mascota AIBO de Sony y un par de brazos robóticos. Además, plantean métricas que permiten medir el desempeño del robot, pero basado en los efectos que produce éste sobre los objetos. De manera similar Lopes y Santos en 2007 [87], plantean dos clases de métricas: nivel de acción y nivel de programa. En la métrica de nivel de acción, se comparan las trayectorias de la mano del robot vistas desde las cámaras del robot, contra las trayectorias demostradas de la mano humana transformadas al marco del robot, también vistas desde las cámaras del robot. En cuanto a la métrica de nivel de programa, esta se

basa en la meta de la tarea, por ejemplo ubicar platos en una mesa. La forma en la cual la tarea se resuelve o la velocidad y posturas de la mano no son relevantes. El autor presenta ejemplos de ambas métricas, aunque no con valores numéricos o con gráficas.

Calinon y otros en 2010 [20], plantean cinco métricas dependientes de la cantidad de estados usados en el modelo oculto de Markov, los autores analizan cómo varía la métrica al cambiar el número de estados. En estas se emplea el tiempo y el error medio cuadrático o RMS (del inglés Root mean Square). Las métricas se mencionan a continuación.

- **Error RMS (M1)**: Compara la trayectoria generada y contra las trayectorias demostradas x_m.

  $$M1 = \frac{1}{MT} \sum_{m=1}^{M} \sum_{k=1}^{T} \|y(k) - x_m(k)\|, \tag{2-8}$$

  donde M es la cantidad de trayectorias demostradas y T es la cantidad de tiempo de la demostración.

- **Error RMS con DTW (M2):** Similar a la anterior pero da prioridad a la información espacial.

- **Norma del jerk (M3):** Mide qué tanto suaviza las trayectorias una técnica de aprendizaje.

  $$a = \ddot{y}, \tag{2-9}$$

  $$M3 = \frac{1}{T} \sum_{k=1}^{T} \|\dot{a}\|. \tag{2-10}$$

  En la cual a es la aceleración y $\dot{a}$ es el jerk.

- **Tiempo de Aprendizaje (M4):** Es el tiempo que tarda el sistema en entrenar o calcular la técnica de aprendizaje.

- **Tiempo de cómputo (M5):** Es el tiempo que tarda el sistema en calcular las salidas.

En 2010 Nope y colaboradores [101], plantearon cuatro métricas de gestos humanos, estas se basan en comparar la trayectoria de los gestos con figuras geométricas conocidas, los autores evalúan las métricas propuestas utilizando dos criterios: por encuesta a personas sobre la calidad observada

del gesto, y comparando el gesto ejecutado por el robot con figuras geométricas de la forma de la trayectoria del gesto.

El empleo de métricas tiene la ventaja de poder obtener los errores presentados en la reproducción del robot, con estas incluso se puede emplear aprendizaje por refuerzo para mejorar la trayectoria [52]. El error RMS tiene como ventajas que es fácil y rápido de calcular y como desventaja que para dos trayectorias con el mismo error, la forma de estas podría ser más conveniente una que otra. En este trabajo, se emplean las métricas de error RMS o métrica M1 y el tiempo de aprendizaje o métrica M4, ya que son las más empleadas por los autores en PpD.

2.2. Trabajos Relacionados

A continuación se revisan temas directamente relacionados con este trabajo, como la generación de nuevas trayectorias a partir de las demostradas, la recuperación ante fallos y el aprendizaje incremental. En la generación de nuevas trayectorias a partir de las demostradas, se hace mayor énfasis en la técnica de modelos de mezcla de gaussianas parametrizados en la tarea, ya que esta técnica es luego empleada tanto en la recuperación ante fallos, como extendida para que tenga la capacidad de aprendizaje incremental.

2.2.1. Generación de Nuevas Trayectorias a Partir de las Trayectorias Demostradas

Tomando como ejemplo la ejecución de una trayectoria que lleva un objeto de A a B (Figura **2-7**), si ahora se desea llevarlo a C, la trayectoria modelada con las técnicas vistas en la Sección 2.1.3, no permiten que el robot ahora lleve el objeto de A a C, guardando además una similitud con la trayectoria demostrada. Según la técnica, los trabajos que generan nuevas trayectorias se agrupan en: *i)* sistemas dinámicos, *ii)* modelo oculto de Markov paramétrico, *iii)* modelo de mezcla de gaussianas parametrizado en la tarea, y *iv)* modificaciones de las primitivas dinámicas de movimiento.

Sistemas Dinámicos

En Hersch y otros [56], se presentó un trabajo donde se generan nuevas trayectorias similares a las demostradas, tanto en el espacio de la tarea como en el de articulación. Con las demostraciones, generan una trayectoria codificada $\boldsymbol{X}^o$ con un modelo GMM en el espacio cartesiano, la técnica genera una nueva trayectoria que varía para lograr llegar a un nuevo punto $\boldsymbol{X}^g$. La variación se logra usando una modulación con un sistema dinámico $\boldsymbol{X}^s$, realizando algunas modificaciones a lo presentado en [56] para facilitar la comprensión, se tiene:

$$\ddot{\boldsymbol{X}}^s(t) = \alpha\left(-\dot{\boldsymbol{X}}^s(t) + \beta(\boldsymbol{X}^g - \boldsymbol{X}^d(t))\right), \tag{2-11}$$

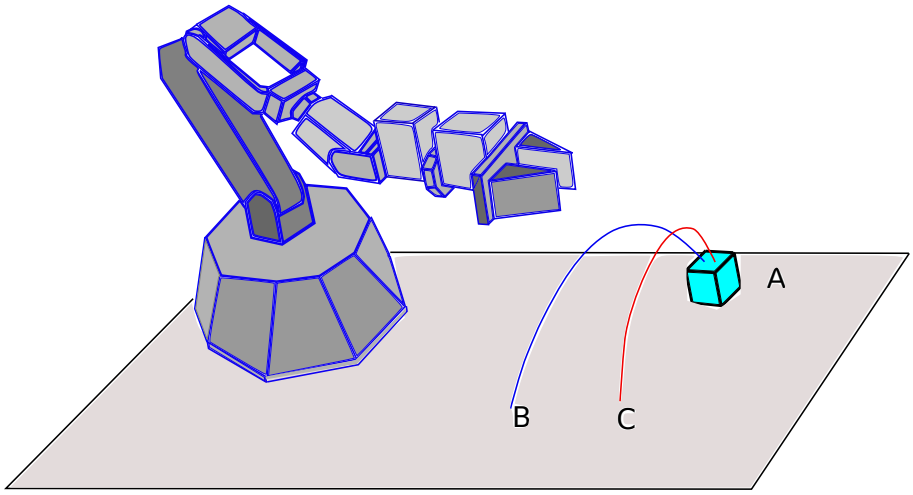


Figura 2-7: Ejemplo de trayectoria demostrada al mover un objeto de la posición A a B (en azul) y la nueva trayectoria generada si se moviera a C (en rojo).

donde $\boldsymbol{X}^d$ es la nueva trayectoria deseada, α y β son las constantes del sistema dinámico. Sumando con una función de ponderación $\gamma(t)$, la respuesta del sistema y la trayectoria codificada, se obtiene la nueva trayectoria $\dot{\boldsymbol{X}}^d(t)$:

$$\dot{\boldsymbol{X}}^d(t) = (\gamma_0 - \gamma(t))\dot{\boldsymbol{X}}^s(t) + \gamma(t)\dot{\boldsymbol{X}}^o(t), \tag{2-12}$$

calculando previamente:

$$\dot{\boldsymbol{X}}^s(t) = \dot{\boldsymbol{X}}^d(t) + \ddot{\boldsymbol{X}}^s(t) \tag{2-13}$$

La función γ decrece en un tiempo T, desde un valor γ_0 hasta un valor de cero, y se define como:

$$\gamma(t) = \gamma_0 max\left(\left(\frac{T-t}{T}\right)^2, 0\right). \tag{2-14}$$

Aunque se genera una nueva trayectoria, solo modifica el tramo final, lo que limita su aplicación a problemas donde se requiere cambiar solo la posición final.

En 2008 Skoglund y otros [123], presentaron una técnica que permite variar la posición inicial del efector final al realizar una tarea de tomar y poner. Cada demostración es codificada con un modelo difuso del tipo Takagi-Sugeno. Para la generalización, la técnica usa modelos dinámicos que se adicionan a la trayectoria codificada, un punto atractor varía la salida final de la adición de los modelos difuso y dinámico. Dado que esta técnica es similar a la técnica presentada en [56], es posible cambiar posiciones iniciales o finales, pero no los puntos intermedios de la trayectoria.

En 2009 Muhling y otros [96], propusieron una técnica que permite evadir obstáculos, basada en puntos atractores y regresión de mezcla de gaussianas. Las demostraciones son codificadas usando modelos de mezcla de gaussianas, con la regresión obtienen la trayectoria que sirve para inicializar la ubicación de los puntos atractores, a partir de un proceso de optimización estos puntos se reubican, dando origen a la nueva trayectoria. La función de optimización se basa en la matriz de covarianza, resultado de la regresión y de requerimientos como los límites de las articulaciones y la ubicación de los obstáculos. Su desventaja es el tiempo que tarda la optimización.

Aprendizaje por Refuerzo

Es una técnica que permite que la respuesta del sistema mejore a partir de reproducciones de la tarea, cada ensayo o reproducción recibe una calificación o recompensa r según el desempeño que tenga y con esta se cambian los parámetros del modelo de la tarea. Para el caso de generación de trayectorias el aprendizaje por refuerzo adapta las trayectorias como se explicará a continuación. Aunque existen varios trabajos que combinan imitación y aprendizaje por refuerzo, se discuten dos de ellos [53] y [70].

En Guenter y otros [53], a partir de las demostraciones $\dot{\xi}^{dem}$ obtienen un modelo de mezcla de gaussianas GMM $\dot{\xi}^{m}$, con este y un sistema dinámico emplean NAC (del Inglés Natural Actor Critic), para modificar los centros de las gaussianas μ que describen las trayectorias en cada ensayo q:

$$\mu_q = \mu_{q-1} + \alpha \boldsymbol{w}_q, \tag{2-15}$$

donde α es una rata de aprendizaje y $\boldsymbol{w}$ es un vector de pesos que depende de la recompensa r, la cual los autores definen como:

$$r_q(\dot{\xi}^r, \xi^r) = \sum_{t=0}^{T} -c_1|\dot{\xi}^r_t - \dot{\xi}^m_{t,q=1}| - c_2|\xi^r_T - \xi^g_T|, \tag{2-16}$$

donde c_1 y c_2 son constantes de ponderación, ξ^r es ruido simulado usado para explorar el espacio solución, $\dot{\xi}^m_{q=1}$ es la respuesta del modelo para el primer ensayo, y ξ^g_T es la posición objetivo $(t = T)$. El primer término trata de maximizar la similitud de la trayectoria generada final y la trayectoria modelada $\dot{\xi}^m_{t,q=1}$, el efecto del segundo término es que r tiende a un máximo cuando se logra la posición objetivo.

En 2010 Peters y Kober [70] emplean primitivas dinámicas de movimiento con aprendizaje por refuerzo para que un robot aprenda a insertar una bola en una copa. Los autores plantean una técnica llamada PoWER (del inglés Policy Learning by Weighting Exploration), la cual no requiere el modelo de la tarea y encuentra la solución con pocos ensayos, este igual que el presentado en [53] actualiza los pesos en función de la recompensa, los autores expresan que esta técnica es mejor a la presentada por Guenter y otros [53], ya que converge en menos ensayos, la actualización de los

parámetros presenta menor varianza y las acciones son perturbadas frecuentemente (ruido en los movimientos del robot).

Modelo Oculto de Markov Paramétrico

El modelo oculto de Markov paramétrico (Abreviado como PHMM del inglés Parametrized HMM), fue propuesto por Wilson y Bobick [134]. Este consiste básicamente en un modelo HMM que tiene parámetros adicionales que modifican la respuesta de salida. Dado un modelo HMM gaussiano compuesto por los parámetros: probabilidades de transición $a_{i,j}$, vectores de salida (promedios de las gaussianas μ_j y covarianzas Σ_j), con j el estado actual e i el estado anterior, le agregan un vector de parámetros adicional $\boldsymbol{\theta}$, que modifica el vector de promedios $\boldsymbol{\mu}$:

$$\hat{\boldsymbol{\mu}}_j(\theta) = \boldsymbol{W}_j\boldsymbol{\theta} + \boldsymbol{\mu}_j, \tag{2-17}$$

donde las columnas de la matriz $\boldsymbol{W}$ transforman un espacio d-dimensional del vector $\boldsymbol{\theta}$, a la dimensión del vector $\boldsymbol{\mu}$. La estimación de los parámetros del modelo PHMM, es muy similar al usado para HMM o sea el algoritmo EM, pero modificado para incluir el parámetro adicional W, para ello definen nuevas variables:

$$\boldsymbol{Z}_j = \begin{bmatrix}\boldsymbol{W}_j & \boldsymbol{\mu}_j\end{bmatrix}, \tag{2-18}$$

$$\boldsymbol{\Omega}_k = \begin{bmatrix}\boldsymbol{\theta}_k \\ 1\end{bmatrix}. \tag{2-19}$$

Con esto $\hat{\boldsymbol{\mu}} = \boldsymbol{Z}\boldsymbol{\Omega}$ y la variable $\boldsymbol{Z}$ es la que se actualiza en el paso de maximización del algoritmo EM.

Herzog y otros [57] y [58], aplican un modelo oculto de Markov paramétrico (PHMM), para la tarea de tomar un objeto. A partir de n modelos ocultos de Markov (HMM) de movimientos similares, en los cuales cambia la posición final del objeto (parámetro), encuentran la nueva trayectoria como una función de combinación de las diferencias entre los n_p parámetros y el nuevo parámetro (nueva ubicación del objeto).

Rozo y otros [114], utilizan PHMM para modificar el movimiento de llenado de un vaso, en este caso el parámetro es la fuerza detectada por un sensor en el efector final (cantidad inicial de líquido). El modelo paramétrico utiliza tres gaussianas para modelar las diferentes trayectorias de articulación final que se producen al variar la fuerza. Según la cantidad inicial de líquido en el vaso (parámetro), el modelo genera la nueva trayectoria de articulación necesaria para llenar el vaso.

La ventaja es su fácil implementación, para el parámetro de la tarea únicamente se requieren posiciones cartesianas, ya que sólo modifican el vector de promedios y no las matrices de covarianzas,

como desventaja, es que las nuevas trayectorias no son tan similares como las que resultan de aplicar otras técnicas que se presentan más adelante. Y además, otra desventaja es que si se estiman en el espacio cartesiano, las trayectorias nuevas llevadas al espacio de articulación podrían ser diferentes a las demostradas en el mismo espacio, debido al problema de múltiples soluciones en la cinemática inversa, para lo cual es necesario emplear alguna técnica, por ejemplo [56], en la que a partir de una trayectoria cartesiana $\boldsymbol{x} = \{x, y, z\}$ y unas trayectorias de articulación demostradas $\boldsymbol{\theta}_d$, se genera una trayectoria de articulación $\boldsymbol{\theta}$, que cumple la trayectoria cartesiana y guarda similitud con $\boldsymbol{\theta}_d$, a través de minimizar un error de similitud.

Modelo de Mezcla de Gaussianas Parametrizado en la Tarea

Basado en el trabajo de Wilson y Bobick [134], Calinon y otros en 2012 [28], presentan el modelo de mezcla de gaussianas parametrizado en la tarea (Abreviado aquí como TPGMM del inglés Task Parametrized Gaussian Mixture Model). De manera similar a PHMM, hay unos parámetros de la tarea (marcos de referencia) y un conjunto de trayectorias demostradas, que se relacionan con estos parámetros. Su ventaja con respecto a PHMM radica en que las trayectorias son más similares a las demostradas, debido a que además del vector de centros, como en PHMM, las matrices de covarianza también son afectadas por los parámetros de la tarea [28]. Por ejemplo en el caso de la tarea de tomar y ubicar como en la Figura **2-7**, los parámetros de la tarea además de posiciones del objeto, tienen información de la orientación de la partida y llegada de las trayectorias de dicha tarea.

Para la estimación del modelo se parte de M demostraciones, las cuales forman un conjunto de datos $\{\boldsymbol{\xi}_n\}_{n=1}^{N}$ de N datos. Denominando a N_p como el número de marcos de referencia, se definen los parámetros de la tarea como: $\{\boldsymbol{A}_{n,j}, \boldsymbol{b}_{n,j}\}_{j=1}^{N_p}$, con $\boldsymbol{A}_{n,j}$ las matrices de orientación y $\boldsymbol{b}_{n,j}$ los vectores de posición. Los índices n y j representan la muestra de tiempo n y el $j-\acute{e}simo$ marco de referencia.

Los parámetros del modelo son $\{\pi_i, \boldsymbol{Z}_{i,j}^{\mu}, \boldsymbol{Z}_{i,j}^{\Sigma}\}$, que representan los coeficientes de mezcla, el vector de centros y las matrices de covarianza para cada marco j y la componente de mezcla i, para $i = 1$ a K, teniendo un modelo de K gaussianas. Con estos parámetros del modelo se calcula el centro resultante $\boldsymbol{\mu}_{n,i}$ y la matriz de covarianza $\boldsymbol{\Sigma}_{n,i}$ de cada componente i, como el producto de las gaussianas linealmente transformadas:

$$\mathcal{N}(\boldsymbol{\mu}_{n,i}, \boldsymbol{\Sigma}_{n,i}) = \prod_{j=1}^{N_P} \mathcal{N}(\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\mu} + \boldsymbol{b}_{n,j}, \boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\Sigma}\boldsymbol{A}_{n,j}^{T}). \tag{2-20}$$

Usando la propiedad del producto de la distribución normal, el centroide y la matriz de covarianza de la ecuación anterior son:

$$\boldsymbol{\mu}_{n,i} = \boldsymbol{\Sigma}_{n,i} \sum_{j=1}^{N_P} (\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\Sigma}\boldsymbol{A}_{n,j}^{T})^{-1}(\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\mu} + \boldsymbol{b}_{n,j}), \tag{2-21}$$

$$\boldsymbol{\Sigma}_{n,i} = (\boldsymbol{A}_{n,j}\boldsymbol{Z}^{\Sigma}_{i,j}\boldsymbol{A}^{T}_{n,j})^{-1}. \quad (2\text{-}22)$$

En las ecuaciones (2-21) y (2-22), los parámetros de la tarea afectan tanto el vector de centros, como la matriz de covarianza. Los parámetros del modelo se estiman iterativamente usando una modificación del procedimiento EM [28], el cual tiene en cuenta los parámetros de la tarea.

Los parámetros de la tarea también pueden tener valores de fuerzas como en [116], donde se muestra la solución a una tarea de colocar una pata a una mesa en una cooperación humano-robot. El humano transfiere por demostración, comportamientos en los cuales el robot cambia su rigidez de acuerdo con la posición de los objetos que se están colocando y las fuerzas de cada momento de la tarea, estos dos: posición y fuerza, son los parámetros de la tarea. Tampoco es necesario contar con todos los marcos de referencia, como en [4], donde aún ante falta de uno o varios parámetros de la tarea en una situación dada, es posible el cálculo de las ecuaciones, considerando los parámetros de la tarea existentes en dicha situación.

Por último, en [23] se muestra una comparación de la técnica de TPGMM, con otras estrategias que permiten generalizar trayectorias. Para la comparación agrupa en tres las técnicas de contraste: *i)* estrategias que emplean M modelos para M demostraciones, *ii)* múltiples marcos de referencia, *iii)* codificado en un solo modelo. A través de resultados gráficos y cuatro métricas de un experimento que consiste en amasar una pizza, demuestra que la técnica de TPGMM presenta ventajas sobre las otras técnicas.

La ventaja es la similitud de las nuevas trayectorias con las demostradas, como desventajas están: *i)* Estimación del término de orientación en el parámetro de la tarea; *ii)* Requieren múltiples demostraciones; *iii)* Requieren cálculos adicionales para el espacio de articulación.

Modificaciones de las Primitivas Dinámicas de Movimiento

Las modificaciones a la teoría de DMP permiten que se genere una nueva trayectoria ante cambio de uno o varios parámetros, algunos trabajos son: evasión de obstáculos por campos potenciales [106], manipulación del estilo del movimiento humano [89], modificación de punto inicial y final de trayectorias [107], acople de percepción y aprendizaje por refuerzo [71], generación de nuevas trayectorias [133].

Park y otros [106], plantearon la adicción de campos de potencial a la técnica de DMP, como un método que le permite al robot evadir obstáculos estáticos y dinámicos. La teoría de campos de potencial en robótica varía la trayectoria y velocidad del efector final, al acercarse a un objeto (efecto de repulsión), su desventaja es que considera sólo evasión de obstáculos esféricos.

En Pastor y otros [107], se presentó una modificación a DMP que consiste en variaciones leves de

ésta. La técnica permite cambios en el punto inicial y final de una sola trayectoria original, también agregan términos a DMP para la evasión de obstáculos. Su ventaja es la simplicidad, sus desventajas es que no generaliza con información de múltiples trayectorias y que no permite parámetros de la tarea intermedios en la trayectoria.

Matsubara y otros [89], presentaron una variante en la cual es posible "almacenar" múltiples estilos del movimiento humano en primitivas dinámicas de movimiento, a las ecuaciones del DMP clásico le agregan un vector de parámetros adicional al que llaman estilo. El aporte del autor es que el entrenamiento de la DMP se realiza con múltiples demostraciones, pero variando el estilo o forma de estas. La ventaja es que permite cambiar la meta final de la trayectoria y el estilo de esta, y su desventaja que solo permite un parámetro relacionado con la tarea.

Kober y otros [71], presentaron una modificación en la cual es posible tener señales de entrada que se calculan a partir de un sistema de visión estéreo, dichas señales de entrada permiten que a través de aprendizaje por refuerzo el robot continúe mejorando su desempeño a partir de un aprendizaje por DMP previo. Sus ventajas son que plantean ecuaciones de DMP que codifican en conjunto las articulaciones, y que se pueden tener uno o varios parámetros de la tarea. Como desventajas se tiene que requiere para el aprendizaje por refuerzo de una función de recompensa escrita de manera especifica para el problema, y que no todas las tareas se pueden refinar realizando pruebas de ensayo-error directamente con el robot.

En 2010 Ude y otros [133], presentan una técnica que permite generar nuevas trayectorias similares a un conjunto de trayectorias demostradas, la técnica utiliza DMP y regresión local ponderada del conjunto de trayectorias, para obtener los parámetros óptimos de DMP que permiten generar la nueva trayectoria. En [48] los mismos autores presentan una mejora en el tiempo de estimación al trabajar con modelos DMP, en vez de usar el conjunto de trayectorias para la obtención de los parámetros óptimos, usan modelos de estas obtenidos con procesos de regresión gaussiana GPR. Una de las ventajas de las técnicas anteriores [133] y [48], es que resuelven el problema de generalización tanto en el espacio de la tarea como en el de articulaciones, y su desventaja es que requieren un buen número de demostraciones.

Métricas Usadas en Generación de Nuevas Trayectorias

Cuando se realizan experimentos reales generalmente no se cuenta con trayectorias de contraste, para comparar la trayectoria generada. Una solución al problema de la falta de trayectorias de contraste, es tener conjuntos de trayectorias demostradas de estimación y prueba. Herzog y otros [58], usaron una tarea que consiste en tomar un objeto de una posición en una mesa, los autores tienen una cuadrícula formada por múltiples posiciones del objeto, con lo que se genera una superficie de error, la cual les sirve para analizar las ventajas de usar más o menos trayectorias en la estimación.

Ude y otros [48], utilizaron la técnica de sacar una de las trayectorias demostradas usada para la estimación, para re-estimar el modelo sin ésta y luego calcular el error entre la generalizada y la demostrada, esto lo realizan para todas las trayectorias y obtienen el error promedio. Aunque lo usan para obtener el número de elementos del modelo (funciones base), podría ser usado como una métrica para evaluar la calidad de las trayectorias generadas.

Calinon y otros en el 2013 [23], utilizaron cuatro métricas para comparar tres técnicas que permiten generar nuevas trayectorias (múlti-trayectoria, PGMM y GPR), la primera es el error RMS como se presento anteriormente en la Ecuación 2-8, comparando una demostración existente con la generada para los mismos parámetros de la tarea, la segunda es la probabilidad posterior del modelo $L(\Theta)$. A partir de las ecuaciones presentadas anteriormente para TPGMM:

$$L(\Theta) = \sum_{n=1}^{N} p(\hat{\boldsymbol{\xi}}_n), \tag{2-23}$$

$$p(\hat{\boldsymbol{\xi}}_n) = \sum_{i=1}^{K} \pi_i \mathcal{N}(\hat{\boldsymbol{\xi}}_n | \boldsymbol{\mu}_{n,i}, \boldsymbol{\Sigma}_{n,i}), \tag{2-24}$$

donde $\hat{\boldsymbol{\xi}}$ son los datos de las trayectorias estimadas con GMR y $\Theta = \{\pi, \boldsymbol{Z}^{\mu}, \boldsymbol{Z}^{\Sigma}\}$.

La tercera métrica es el tiempo de estimación que toma cada técnica, la cuarta y última es una métrica específica de la tarea usada en las pruebas, y mide que tanto extrapolan las técnicas comentadas, los autores la llaman calidad de elongación. Como la tarea consiste en extender una masa de pizza con un rodillo, esta métrica, a partir de procesamiento de imágenes obtiene los ejes mayor a_{mayor} y menor a_{menor} de la elipse que forma la masa:

$$e_c = a_{mayor} - a_{menor}, \tag{2-25}$$

donde e_c, es la calidad de elongación. La métrica refleja el requerimiento de extender la masa en una dirección determinada. Los autores concluyen que la técnica de TPGMM es la que logra mejores resultados.

2.2.2. Recuperación Ante Fallos de Ejecución

En el momento de ejecutar una tarea, se pueden presentar fallos de ejecución, por ejemplo cuando un robot coloca la manga de una camisa a una persona, ésta puede presentar enganches con la extremidad humana. Es en este punto, donde es conveniente que el robot pueda realizar una recuperación del fallo, como por ejemplo regresar por la misma trayectoria de colocación y repetir la colocación.

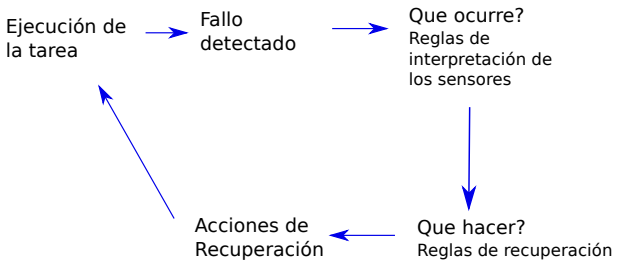


Figura 2-8: Esquema general de recuperación a fallo de ejecución.

Uno de los primeros trabajos en robótica sobre recuperación de fallos es el presentado por Gini y otros [50]. Propusieron una definición del entorno del robot que permite que conozca de antemano las pre-condiciones y pos-condiciones de ejecutar una acción, con lo anterior, construyen una base de datos con las condiciones o reglas de los sensores y actuaciones que llevan a recuperar el error (Figura **2-8**).

En PpD una técnica clásica de caminado bípedo fue mejorada con demostraciones en [91], a partir de reproducir caminatas de un robot NAO programado manualmente con una técnica clásica de caminado, el humano que lo observa suministra acciones correctivas en lazo abierto. Las acciones son almacenadas para luego, a través de un algoritmo, sintonizar a partir de las demostraciones y las señales de los sensores internos del robot, una matriz de ganancias, la cual luego permite que el robot camine en lazo cerrado. Los autores muestran que su técnica aumenta la distancia que el robot puede caminar sin caerse. Si bien el caminado es continuo, las acciones correctivas son dadas al momento de ejecutar la tarea y la corrección se presenta en futuras caminatas y no como respuesta a un fallo. Tampoco hay posibilidad de variaciones de la trayectoria del caminado.

En Mericli y otros [92] y [93], se presentó un sistema que permite adicionar demostraciones correctivas a un algoritmo existente que fue codificado manualmente. El algoritmo inicial se construyó usando una máquina de estados finitos jerárquica. Las demostraciones correctivas pueden ser de dos formas: acciones alternas o modificación de la selección de una acción. Una vez agregadas las demostraciones correctivas, el robot decide si ejecuta la acción del algoritmo o la acción de demostración, basado en la similitud del estado almacenado durante la demostración y el estado actual. El sistema es aplicado para enseñar a un humanoide a driblar en el deporte de fútbol.

En Chang y otros [32], se presentó una técnica que emplea redes de Petri y primitivas dinámicas de movimiento, la cual permite que un robot aprenda por demostración a recuperarse de un error. La técnica adiciona una nueva subred cuando detecta una diferencia entre el estado (pose) actual y el esperado, de varios objetos. La nueva subred es generada a partir de una demostración por parte del maestro, de cómo corregir el error, y la realiza debido a una petición del robot. La técnica se aplica en tareas de tomar, mover y apilar bloques de madera. El sistema no responde ante variaciones de

posición de los objetos.

En 2014 Martinez y otros [88], programaron un robot simulado para realizar un ensamble. Los autores afirman que el sistema es robusto a fallos de ejecución (Ej. partes no identificadas, errores de inserción), gracias al aprendizaje por refuerzo, y que ante un inconveniente, el sistema hace requerimiento al usuario para aprender a solucionarlo.

2.2.3. Aprendizaje Incremental

El aprendizaje incremental permite que se agregue nueva información de la tarea al modelo. Se puede dividir en aprendizaje incremental de tareas y aprendizaje incremental de habilidades (skill). Una habilidad describe una acción básica como por ejemplo operaciones de manipulación o transporte, una tarea es una secuencia de habilidades.

Trabajos que permiten incrementar habilidades son, por ejemplo, el de Calinon y otros 2007 [24], donde se presenta una técnica que permite agregar nuevas trayectorias $\boldsymbol{\xi}$, a un modelo GMM existente $\boldsymbol{\Theta} = \{\boldsymbol{\alpha}, \boldsymbol{\nu}, \boldsymbol{C}\}$, con $\boldsymbol{\alpha}$ los coeficientes de mezcla, $\boldsymbol{\nu}$ vector de centros, y $\boldsymbol{C}$ la matriz de covarianza de cada gaussiana. El autor propone dos tipos de estrategias, la directa y la generativa. La estrategia directa se basa en que las nuevas trayectorias son similares a las anteriores usadas para calcular el modelo, con lo cual el algoritmo EM de ajuste del modelo se ejecuta una vez más o se actualiza $(t+1)$, pero usando la o las nuevas trayectorias, así:

1. *Paso E:* Cálculo de la esperanza $\tilde{E}_i$, a partir del modelo existente y las nuevas trayectorias.

$$\tilde{\mathcal{P}}_{i,n}^{t+1} = \frac{\alpha_i \mathcal{N}(\boldsymbol{\xi}|\boldsymbol{\nu}_i, \boldsymbol{C}_i)}{\sum_{l=1}^{K} \alpha_l \mathcal{N}(\boldsymbol{\xi}|\boldsymbol{\nu}_l, \boldsymbol{C}_l)}, \tag{2-26}$$

$$\tilde{E}_i^{t+1} = \sum_{n=1}^{N} \tilde{\mathcal{P}}_{i,n}^{t+1}, \tag{2-27}$$

   con K el número de gaussianas y N la cantidad de muestras de una trayectoria.

2. *Paso M:* Maximización de la esperanza.

$$\boldsymbol{\Theta}_{t+1} = \mathcal{F}(\boldsymbol{\Theta}_t, \tilde{E}_i^{t+1}, \boldsymbol{\xi}), \tag{2-28}$$

donde $\mathcal{F}$ es el conjunto de ecuaciones para maximizar la esperanza. En la generativa a partir del modelo, se genera una trayectoria aumentada $\boldsymbol{\xi}_g$, con puntos aleatorios en todo su trayecto de acuerdo a la matriz de covarianza:

$$\boldsymbol{\xi}_g = gmr(\boldsymbol{\Theta}_0) + [rand(n)]\boldsymbol{C}, \tag{2-29}$$

que se juntan con los de las nuevas trayectorias y se estima un nuevo modelo $\mathbf{\Theta}_1$, con el algoritmo EM:

$$\boldsymbol{\xi}_c = [\boldsymbol{\xi}_g, \boldsymbol{\xi}], \tag{2-30}$$

$$\mathbf{\Theta}_1 = EM(\boldsymbol{\xi}_c, \mathbf{\Theta_0}). \tag{2-31}$$

En la técnica generativa existe, además, una relación de aprendizaje, la cual permite que se olvide un poco las trayectorias pasadas. Similar a la técnica generativa de Calinon, Lee y Ott en 2011 [78] generan una sola trayectoria a partir del modelo pasado, con esta y la nueva trayectoria recalculan un modelo HMM. La diferencia es que no emplean puntos aleatorios dentro de los límites de covarianza.

Otros autores han abordado el problema del aprendizaje incremental de múltiples habilidades [75], [29] y [81]. Emplean un modelo para cada habilidad y un clasificador (árbol de decisión o una técnica de agrupamiento), con lo cual clasifican la nueva trayectoria que se desea incrementar a la que corresponde con el modelo más similar. Este modelo es estimado nuevamente como en la técnica generativa. Kulic y otros [75], usaron modelos ocultos de Markov factorial. Lee y otros [81], utilizan PCA y modelos de mezcla de gaussianas GMM. Cederborg y otros [29], regresión de mezclas gaussianas. Aunque estas técnicas funcionan para múltiples habilidades, no tratan el tema de generalización de nuevas trayectorias.

En aprendizaje incremental de tareas han sido presentado trabajos como: Niekum y otros [100], Pardowitz y otros [105], Demiris y otros [80]. Niekum y otros [100], presentaron un sistema que aprende de manera incremental a tomar la pata de la mesa de otras ubicaciones y orientaciones, adicionan ramas a un autómata de estados finitos, las cuales luego son condensadas con las ramas iniciales en sus tramos similares.

Pardowitz y otros [105], a partir de varias demostraciones de una tarea, generaron un grafo de precedencia de la tarea, este se compone de las etiquetas de las subtareas y relaciones de precedencia. Dada una demostración adicional, el sistema incrementa el conocimiento de la tarea, actualizando el grafo de manera que se hace más general. Demiris y otros [80], emplean el aprendizaje incremental para aprender tareas complejas a partir de acciones básicas. En un primer estado, el sistema es entrenado con un conjunto de acciones básicas, luego el sistema aprende secuencias de estas acciones básicas al estimar incrementalmente un modelo estocástico de la secuencia. El sistema aprende la secuencia de una tarea, a partir de encontrar la acción más apropiada para un tipo de objeto, esta es encontrada con base en probabilidades obtenidas que relacionan objetos y acciones.

2.3. Resumen

En este capítulo se presentaron de manera acotada pero suficiente, los temas de programación por demostración de robots. En la primera parte se discutió sobre la adquisición y adecuación de la información. Además, siguiendo la forma de clasificación en PpD basada en la complejidad de la tarea (elementales y complejas), se muestran los procesos para su programación, las tareas elementales se modelan generalmente con alguna técnica de aprendizaje de máquina, mientras que las tareas complejas requieren de una segmentación o partición en subtareas, para luego realizar una representación simbólica que permita su reproducción. Otro tema importante, tratado en esta primera parte del capítulo, es el relacionado con las métricas usadas en aprendizaje por demostración. Una de las métricas más empleadas es el error RMS, la cual es fácil de calcular, aunque para dos trayectorias con el mismo valor de error, una puede ser más similar a las demostraciones que otra. Las métricas, también pueden servir para mejorar las reproducciones, usando aprendizaje por refuerzo como en [52]. En las simulaciones e implementaciones de este trabajo, se utilizaron las métricas de error RMS y tiempo de aprendizaje o estimación.

En la segunda parte, se presentaron trabajos que están más relacionados con esta tesis, el problema de la generación de nuevas trayectorias a partir de otras demostradas, ha sido resuelto encontrando técnicas que relacionan parámetros de la tarea con el modelo de las demostradas. Estas técnicas, en sus inicios consistían en usar una función relacionada con una posición final y mezclar su respuesta con la trayectoria demostrada, su desventaja es que solo era para una posición final y una o varias demostraciones pero de una misma posición y cercana a la deseada. Por lo que luego se emplean los modelos estadísticos como HMM, GMM, GPR, pero adicionando parámetros relacionados con la posición final, con estos ya es posible tener varias demostraciones de diferentes posiciones finales demostradas. Las métricas en el caso de generación de nuevas trayectorias, requieren el contraste de la nueva trayectoria con una demostrada o métricas específicas construidas según la tarea. Los otros dos temas tratados, son la recuperación ante fallos de ejecución, donde se presentaron los trabajos existentes hasta el momento y en los que no se aborda el problema de generar nuevas trayectorias y además recuperarse de un fallo, y el de aprendizaje incremental, donde al modelo de la tarea se le puede agregar nueva información, para incluir nuevos comportamientos por parte del robot.

3 Técnicas de Aprendizaje: Funcionamiento y Discusión

En esté capítulo se presentan las técnicas de aprendizaje en que está basada esta tesis, se usaron modelos de mezcla de gaussianas parametrizados en la tarea, los cuales permiten modelar varias trayectorias demostradas, que además de los parámetros del modelo, pueden ser parametrizados con marcos de referencia relacionados con la tarea. También se empleó una red neuronal llamada máquina de aprendizaje extremo, la cual se aplica como una técnica para modelar las trayectorias y también para aprender el modelo cinemático directo de un brazo de robot. Estas como se presentarán en un capítulo posterior se emplean en conjunto, como una forma de resolver el problema de generación de nuevas trayectorias a partir de las demostradas.

En la Sección 3.1 se presentan técnicas que permiten incrementar el aprendizaje de modelos de mezcla de gaussianas, en la Sección 3.2 se presenta una introducción al modelo oculto de Markov paramétrico, luego, en la Sección 3.3, se presenta la técnica de modelo de mezcla de gaussianas parametrizado en la tarea, en la Sección 3.4 se presenta la máquina de aprendizaje extremo, y por último se resume el capítulo.

3.1. Técnicas de Aprendizaje Incremental en GMM

Dado un modelo GMM existente, estas técnicas permiten agregar nuevas trayectorias, generando un nuevo modelo, sin necesidad de re-estimar el modelo con todas las trayectorias. Cuando se estima el modelo con todas las trayectorias se denomina estimación en lote. Las técnicas son: *i)* Generativa, *ii)* Directa, *iii)* Adición de modelos.

En un modelo GMM compuesto por K gaussianas (estados), cada una de ellas con parámetros: α_i coeficientes de mezcla, $\boldsymbol{\nu}_i$ vector de centros y $\boldsymbol{C}_i$ matriz de covarianza de la *i-ésima* gaussiana, modela múltiples trayectorias de demostración similares y que tardan el mismo tiempo. Este modelo es estimado con el algoritmo EM [24] y a partir de los parámetros del modelo, se puede luego recuperar la trayectoria usando regresión de mezcla de gaussianas GMR [25]. Lo que se describe a continuación.

Como cada trayectoria demostrada se compone del tiempo t, y las D variables de posición $\boldsymbol{y}$, los parámetros del modelo $\{\alpha_i, \boldsymbol{\nu}_i, \boldsymbol{C}_i\}$, se pueden descomponer en:

$$\boldsymbol{\nu}_i = \begin{bmatrix} \nu_i^t \\ \nu_i^{\boldsymbol{y}} \end{bmatrix}, \tag{3-1}$$

$$\boldsymbol{C}_i = \begin{bmatrix} C_i^t & C_i^{t\boldsymbol{y}} \\ C_i^{\boldsymbol{y}t} & C_i^{\boldsymbol{y}} \end{bmatrix}. \tag{3-2}$$

Para cada componente i, la distribución esperada ξ_i^y, dado un dato de tiempo ξ_t es:

$$\hat{\xi}_i^y = \nu_i^y + C_i^{yt}(C_i^t)^{-1}(\xi_t - \nu_i^t), \tag{3-3}$$

y la covarianza condicionada dado ξ_t es:

$$\hat{C}_i^y = C_i^y - C_i^{yt}(C_i^t)^{-1}C_i^{ty}. \tag{3-4}$$

A partir de esta, se calcula el valor de respuesta:

$$\hat{\xi}_y = \sum_{i=1}^{K} \beta_i \hat{\xi}_i^y, \tag{3-5}$$

donde:

$$\beta_i = \frac{\alpha_i \mathcal{N}(\xi_t | \nu_i^t, C_i^t)}{\sum_{l=1}^{K} \alpha_l \mathcal{N}(\xi_t | \nu_l^t, C_l^t)}. \tag{3-6}$$

Y la matriz de covarianza de ξ_y, es:

$$\hat{C}^y = \sum_{i=1}^{K} \beta_i^2 \hat{C}_i^y. \tag{3-7}$$

3.1.1. Método Generativo

Propuesto por Calinon [24], se basa en que partir de un modelo inicial $\boldsymbol{\theta}_0 = \{\alpha, \boldsymbol{\nu}, \boldsymbol{C}\}$, se utiliza regresión de mezcla de gaussianas GMR, para generar un conjunto de datos ξ_g, en este caso no solo se genera una trayectoria respuesta, sino que además se generan puntos aleatorios alrededor de la trayectoria, usando la matriz de covarianza $\hat{C}$, que se calcula con (3-7).

$$\hat{\xi}_y = gmr(\theta_0, \xi_t), \tag{3-8}$$

$$\hat{\xi}_g = \hat{\xi}_y + rand(n)eig(\hat{C}^y), \tag{3-9}$$

donde $rand(n)$ genera un vector de n valores aleatorios, y eig son los valores propios de $\hat{C}^y$. A los datos generados se les agrega los datos de las nuevas trayectorias ξ_n, formando un nuevo conjunto, el cual es utilizado para estimar un nuevo modelo GMM con el algoritmo EM:

$$\boldsymbol{\xi}_c = [\boldsymbol{\xi}_g, \boldsymbol{\xi}_n], \tag{3-10}$$

$$\theta = EM(\xi_c, \theta_0). \tag{3-11}$$

Además de lo anterior, Calinon propuso una relación entre la cantidad de datos generados con la cantidad de datos de la nueva trayectoria, lo que permite un balance entre la cantidad de nuevos datos aprendidos y la cantidad de datos antiguos retenidos. Para esto plantea el uso de una rata de aprendizaje α entre cero y uno, la cual disminuye a medida de que se van agregando trayectorias al modelo:

$$\alpha_1 = \frac{\alpha_0}{\alpha_0 + 1}, \tag{3-12}$$

sea n_1 la cantidad de puntos que se duplica por cada punto de la trayectoria nueva, en cada iteración este valor se reduce con:

$$n_1 = |\alpha_1 n|, \tag{3-13}$$

el símbolo $|\cdot|$ es la aproximación al mayor valor entero, con n la cantidad de puntos que se generan por el conjunto completo. La cantidad de puntos por punto de las trayectorias generadas es igual a:

$$n_2 = n - n_1, \tag{3-14}$$

Con lo anterior se modifica la Ecuación 3-10, con las nuevas cantidades de datos generados y nuevos, logrando un balance entre ellos.

3.1.2. Método Directo

Esta técnica se basa en separar en las ecuaciones del algoritmo EM, las componentes correspondientes a las trayectorias iniciales y las trayectorias que se desean agregar. La separación parte de la base de que el conjunto de probabilidades posteriores se mantiene igual tanto para las trayectorias pasadas, como para las nuevas. Con las ecuaciones, el modelo GMM se actualiza de manera directa a partir de las trayectorias que se desean incrementar. Dado el modelo GMM anterior $\{\boldsymbol{\alpha}, \boldsymbol{\nu}, \boldsymbol{C}\}$, las probabilidades posteriores $\mathcal{P}_{i,n}$ de las trayectorias pasadas y las nuevas trayectorias $\tilde{\boldsymbol{\xi}}$, el modelo se actualiza con el algoritmo EM, pero modificado con la separación de los componentes del método directo [24]:

Paso E: Cálculo de la esperanza a partir de las probabilidades pasadas:

$$\tilde{\mathcal{P}}_{i,n}^{t+1} = \frac{\alpha_i \mathcal{N}(\tilde{\boldsymbol{\xi}}|\boldsymbol{\nu}_i, \boldsymbol{C}_i)}{\sum_{l=1}^{K} \alpha_l \mathcal{N}(\boldsymbol{\xi}|\boldsymbol{\nu}_l, \boldsymbol{C}_l)}, \tag{3-15}$$

$$\tilde{E}_i^{t+1} = \sum_{n=1}^{N} \tilde{\mathcal{P}}_{i,n}^{t+1}. \tag{3-16}$$

Paso M:

$$\begin{aligned}
\alpha_i^{t+1} &= \frac{E_i + \tilde{E}_i^{t+1}}{N + \tilde{N}}, \\
\nu_i^{t+1} &= \frac{E_i \nu_i + \sum_{n=1}^{\tilde{N}} \tilde{\mathcal{P}}_{i,n}^{t+1} \xi_n}{E_i + \tilde{E}_i^{t+1}}, \\
C_i^{t+1} &= \frac{E_i (C_i + (\nu_i - \nu_i^{t+1})(\nu_i - \nu_i^{t+1})^T)}{E_i + \tilde{E}_i^{t+1}} + \frac{\sum_{n=1}^{\tilde{N}} \tilde{\mathcal{P}}_{i,n}^{t+1} (\xi_n - \nu_i^{t+1})(\xi_n - \nu_i^{t+1})^T}{E_i + \tilde{E}_i^{t+1}}.
\end{aligned} \tag{3-17}$$

3.1.3. Adición de Modelos

Esta técnica permite adicionar los parámetros de un modelo existente, estimado con unas trayectorias iniciales, a los parámetros del modelo estimado de las trayectorias nuevas. Se basa en la adición de modelos de mezcla de gaussianas (GMM) propuesto en el 2004 por Hall y Hikcs [54]. La técnica consta de los siguientes pasos:

1. Se calcula el modelo de la o las nuevas trayectorias.
2. Se concatena el modelo obtenido en el paso anterior con el modelo existente.
3. Se simplifica el modelo resultado de la concatenación, usando un proceso de optimización.

Concatenación:
Dados dos modelos GMM $\{\alpha_i, \boldsymbol{\nu}_i, \boldsymbol{C}_i\}$, $\{\grave{\alpha}_i, \grave{\boldsymbol{\nu}}_i, \grave{\boldsymbol{C}}_i\}$ con probabilidad de distribución:

$$\mathcal{P}_1(\boldsymbol{\xi}) = \sum_{i=1}^{K_1} \alpha_i \mathcal{N}(\boldsymbol{\xi}|\boldsymbol{\nu}_i, \boldsymbol{C}_i), \tag{3-18}$$

$$\mathcal{P}_2(\boldsymbol{\xi}) = \sum_{i=1}^{K_2} \grave{\alpha}_i \mathcal{N}(\boldsymbol{\xi}|\grave{\boldsymbol{\nu}}_i, \grave{\boldsymbol{C}}_i), \tag{3-19}$$

para $j = 1, ..., N_P$ marcos de referencia, la suma de las probabilidades es:

$$\begin{aligned}
\mathcal{P}_s(\boldsymbol{\xi}) &= f_1\mathcal{P}_1(\boldsymbol{\xi}) + f_2\mathcal{P}_2(\boldsymbol{\xi}), \\
\mathcal{P}_s(\boldsymbol{\xi}) &= f_1\sum_{i=1}^{K_1}\alpha_i\mathcal{N}(\boldsymbol{\xi}|\boldsymbol{\nu}_i, \boldsymbol{C}_i) + f_2\sum_{i=1}^{K_2}\grave{\alpha}_i\mathcal{N}(\boldsymbol{\xi}|\grave{\boldsymbol{\nu}}_i, \grave{\boldsymbol{C}}_i), \\
\mathcal{P}_s(\boldsymbol{\xi}) &= \sum_{i=1}^{K_1+K_2}\beta_i\mathcal{N}(\boldsymbol{\xi}|\boldsymbol{\vartheta}_i, \boldsymbol{\Xi}_i).
\end{aligned} \tag{3-20}$$

Las variables $\boldsymbol{\beta}$, $\boldsymbol{\vartheta}$, $\boldsymbol{\Xi}$ estan definidas de la siguiente manera:

$$\begin{aligned}
\boldsymbol{\beta} &= [f_1\boldsymbol{\alpha}, f_2\grave{\boldsymbol{\alpha}}], \\
\boldsymbol{\vartheta} &= [\boldsymbol{\nu}, \grave{\boldsymbol{\nu}}], \\
\boldsymbol{\Xi} &= [\boldsymbol{C}, \grave{\boldsymbol{C}}]
\end{aligned} \tag{3-21}$$

Que equivale al modelo concatenado $\{\beta_i, \boldsymbol{\vartheta}_i, \boldsymbol{\Xi}_i\}$, el cual tiene $(K_1 + K_2)$ componentes, donde $f_2 = 1 - f_1$.

Simplificación:
Dado los elementos del modelo concatenado, se puede simplificar a un modelo con K componentes, donde $K < K_1 + K_2$. La simplificación se realiza a partir de una matriz de mezcla, entre el modelo concatenado y el simplificado $\boldsymbol{w}$:

$$\pi_l = \sum_{i=1}^{K_1+K_2} w_{i,l}\beta_i, \tag{3-22}$$

$$\boldsymbol{\mu}_l = \frac{1}{\pi_l}\sum_{i=1}^{K_1+K_2} w_{i,l}\beta_i\boldsymbol{\vartheta}_i, \tag{3-23}$$

$$\boldsymbol{\Sigma}_l = \frac{1}{\pi_l}\left(\sum_{i=1}^{K_1+K_2} w_{i,l}\beta_i(\boldsymbol{\Xi}_i + \boldsymbol{\vartheta}_i(\boldsymbol{\vartheta}_i)^T\right) - \boldsymbol{\mu}^{(j)_l}(\boldsymbol{\mu}_l)^T. \tag{3-24}$$

Con las restricciones:

$$\sum_{i=1}^{K_1+K_2} w_{i,j} = 1, \tag{3-25}$$

$$\sum_{i=1}^{K_1+K_2} w_{i,j}\alpha_i < 1. \tag{3-26}$$

Contrario al algoritmo de optimización descrito en [54], para hallar la matriz de mezcla $\boldsymbol{w}$, que presenta una convergencia lenta, se utilizó una función de optimización basada en el error entre las trayectorias generadas por el modelo concatenado y el modelo simplificado. A partir de un $\boldsymbol{w}$

inicial y dicha función que se desea minimizar, se hallan los pesos que permiten obtener el modelo simplificado. El valor inicial de $\boldsymbol{w}$, se seleccionó de manera heurística, como:

$$\boldsymbol{w} = 0{,}45 \begin{bmatrix} I_{K_1 \times K} \\ I_{K_2 \times K} \end{bmatrix}, \tag{3-27}$$

donde I es la matriz identidad.

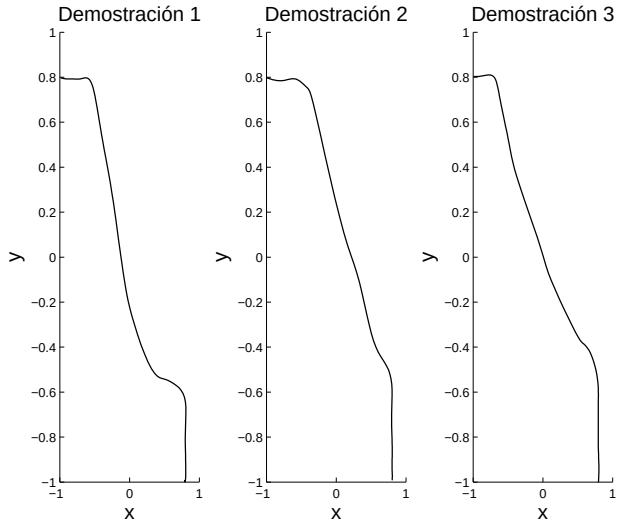
La función a minimizar que se utilizó está dada por:

$$f = \frac{1}{T_m} \sum_{m=1}^{M} \sum_{k=1}^{T_m} \|\boldsymbol{y}_m(k) - \boldsymbol{y}_m^s(k)\|^2, \tag{3-28}$$

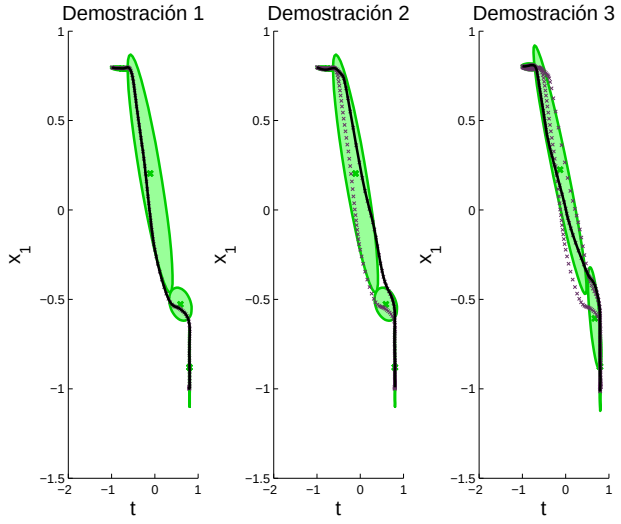
donde $\boldsymbol{y}_m$ es el conjunto formado por las trayectorias reproducidas usando el modelo anterior y las trayectorias demostradas que se desean agregar, y $\boldsymbol{y}_m^s$ son las trayectorias equivalentes, reproducidas con el modelo simplificado a partir de los parámetros de la tarea, y las cuales se recalculan en cada iteración del proceso de minimización. Para la implementación del algoritmo se utilizó la función para Matlab *fminsearchcon*, sujeta a las restricciones descritas en las ecuaciones (6-9) y (6-10).

Ejemplo 3.1:

Dada la trayectoria demostrada 1 en la Figura **3-1**a, se estima un modelo GMM inicial. Con las demostraciones 2 y 3 se incrementa este modelo al aplicar cada uno de los métodos (generativo, actualización directa, adición de modelos), los resultados se presentan en las Figuras **3-1**b y **3-2**. En las figuras, a la izquierda las gaussianas del modelo inicial, estimado con la demostración 1; centro y derecha, la técnica re-estima la ubicación y orientación de las gaussianas al agregar las demostraciones 2 y 3.

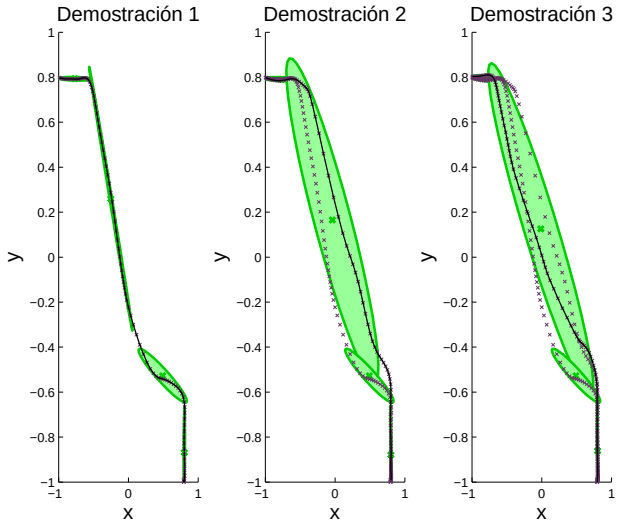


(a) Demostraciones.

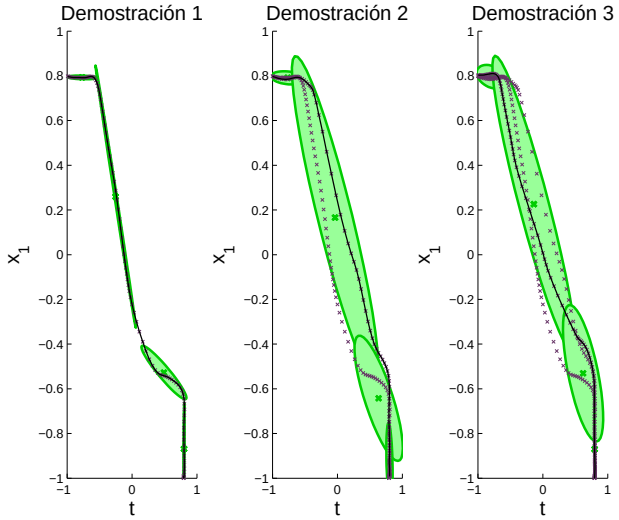


(b) Método generativo.

Figura 3-1: Trayectorias demostradas y resultados al incrementar el modelo inicial con las demostraciones 2 y 3, empleando el método generativo.



(a) método de actualización directa.



(b) método de adicción de modelos.

Figura 3-2: Resultados al incrementar el modelo inicial con las demostraciones 2 y 3 y los métodos de actualización directa y adición de modelos.

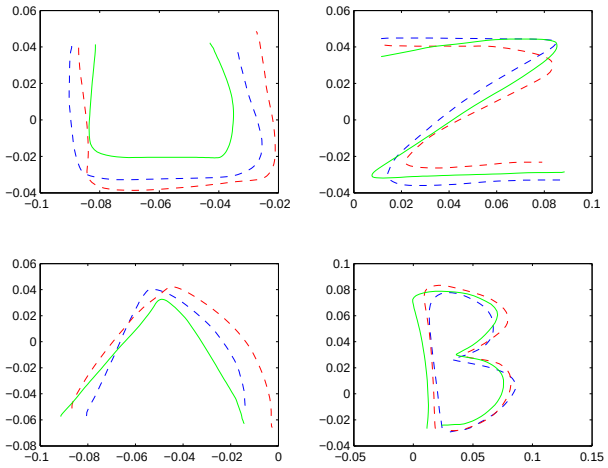


Figura 3-3: Demostraciones de las letras U,Z,A,B (Lineas Azul y roja punteada, demostraciones iniciales, verde y continua demostración a incrementar.

3.1.4. Comparación Experimental de las Técnicas Incrementales en GMM

Usando el error RMS y el tiempo de estimación se compararon las técnicas, para ello se utilizaron las demostraciones de la Figura **3-3**, las cuales corresponden a las letras U, Z, A, B (demostraciones tomadas de [18]). En las Figuras **3-4** y **3-5** se muestran los errores RMS y los tiempos de estimación al incrementar la demostración con línea continua y color verde. La estimación en lote del modelo es la que se realiza con las tres demostraciones. De las figuras **3-4** y **3-5** se puede decir que, dado que presenta el error RMS más cercano a la estimación por lote y el menor tiempo, el método de actualización directa es el mejor.

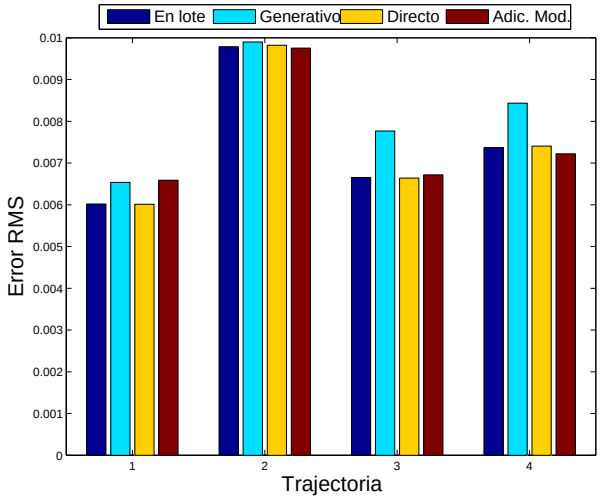


Figura 3-4: Errores RMS de la estimación en lote y las técnicas incrementales en GMM.

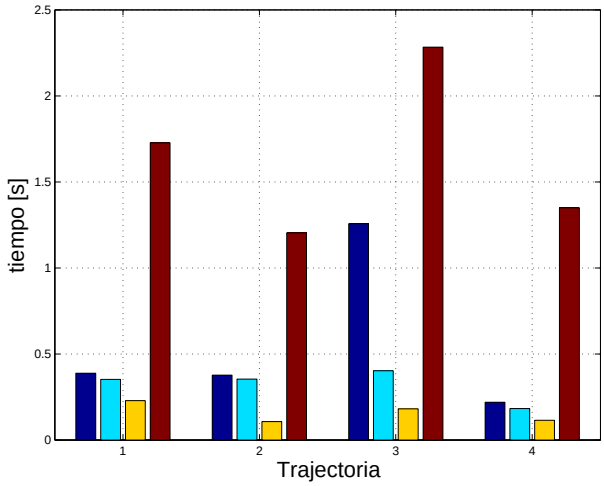


Figura 3-5: Tiempos de estimación en lote y de las técnicas incrementales en GMM.

3.2. Modelo Oculto de Markov Paramétrico

El modelo oculto de Markov paramétrico (abreviado como PHMM del inglés Parametrized HMM), fue propuesto por Wilson y Bobick [134]. Este consiste básicamente en un modelo HMM que tiene parámetros adicionales que modifican la respuesta de salida. Dado un modelo HMM gaussiano compuesto por los parámetros de: probabilidades de transición $a_{i,j}$, vectores de salida (promedios de las gaussianas $\hat{\mu}_j$ y covarianzas Σ_j), con j el estado actual e i el estado anterior, se agrega un vector de parámetros adicional que permiten cambiar el vector de promedios μ:

$$\mu_j(\theta) = W_j\theta + \bar{\mu_j}, \tag{3-29}$$

donde las columnas de la matriz W transforman un espacio d-dimensional del vector θ a la dimensión del vector μ.

La estimación de los parámetros del modelo PHMM, es muy similar al usado para HMM o sea el algoritmo EM, pero está modificado para incluir el parámetro adicional W, para ello definen nuevas variables:

$$Z_j = \begin{bmatrix} W_j & \bar{\mu_j} \end{bmatrix}, \tag{3-30}$$

$$\Omega_k = \begin{bmatrix} \theta_k \\ 1 \end{bmatrix}. \tag{3-31}$$

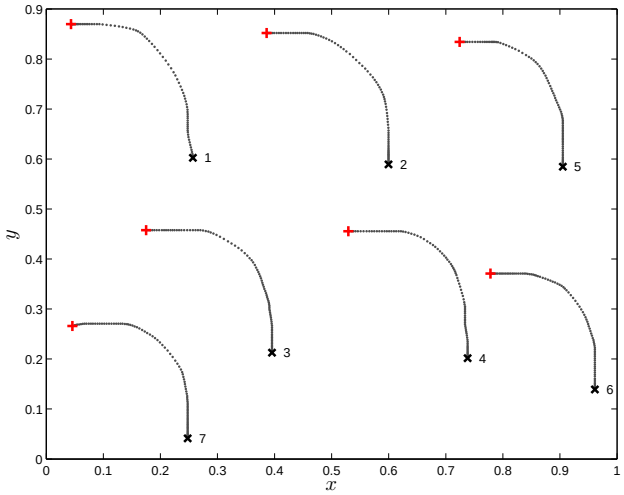
Con esto $\hat{\boldsymbol{\mu}} = \boldsymbol{Z\Omega}$ y la variable $\boldsymbol{Z}$ es la que se actualiza en el paso de maximización. La esperanza del logaritmo de la probabilidad es:

$$Q(\phi'|\phi) = E[log(\mathcal{P}(\boldsymbol{x}, \phi')], \tag{3-32}$$

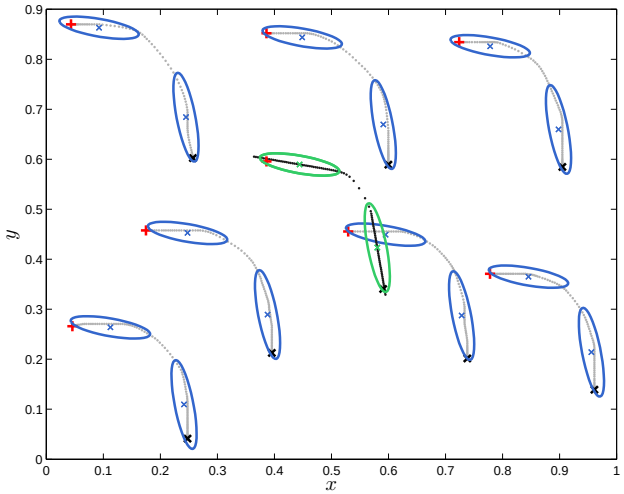
con $\phi = \{W, \mu, \Sigma\}$ los parámetros del modelo previo y ϕ' los que se encuentran con el proceso de maximización. La variable $\boldsymbol{x}$ son los datos conocidos para estimar el modelo, derivando la función de esperanza del logaritmo de la probabilidad del modelo con respecto a Z_j e igualando a cero, se encuentra la ecuación que permite encontrar el valor de Z_j, el procedimiento completo se puede consultar en [134].

Ejemplo 3.2

En la Figura **3-6**a se muestran 7 trayectorias xy demostradas, que permiten estimar un modelo PHMM de dos gaussianas, el inicio y final son el vector de parámetros θ, por ejemplo para la



(a) Demostraciones.



(b) Ejemplo de reproducción de una nueva trayectoria (línea punteada en color negro).

Figura 3-6: Trayectorias demostradas usadas para estimar el modelo PHMM y resultado del modelo PHMM para nuevos parámetros.

trayectoria uno:

$$\theta_1 = \begin{bmatrix} 0{,}0434 \\ 0{,}8697 \\ 0{,}2570 \\ 0{,}6024 \end{bmatrix}. \tag{3-33}$$

Con la información de las trayectorias $\boldsymbol{x}$ y el conjunto de vectores de parámetros $\boldsymbol{\theta}$, se estima ϕ el modelo PHMM con el procedimiento EM modificado:

$$\phi = EM\_modificado(\boldsymbol{x}, \boldsymbol{\phi}_0, \boldsymbol{\theta}), \tag{3-34}$$

donde $\boldsymbol{\phi}_0$ es un valor inicial aleatorio. En la Figura **3-6**b, se muestra un ejemplo de la generación de una trayectoria dado unos valores de inicio y fin:

$$\theta_n = \begin{bmatrix} 0{,}3922 \\ 0{,}5963 \\ 0{,}5907 \\ 0{,}3398 \end{bmatrix}, \tag{3-35}$$

para ello se usa la regresión de mezcla de gaussianas, que da como resultado las gaussianas de color verde y de las cuales se obtiene la trayectoria (línea punteada de color negro).

3.3. Modelo de Mezcla de Gaussianas Parametrizado en la Tarea

Basado en el trabajo de Wilson y Bobick [134], Calinon en 2012 propuso el Modelo de Mezcla de Gaussianas Parametrizado en la Tarea (Abreviado aquí como TPGMM del inglés Task Parametrized GMM) [28]. En la Figura **3-7** se muestran las tres etapas del proceso que permite la generación de nuevas trayectorias. En una primera etapa se adquiere la información de las demostraciones y de los parámetros de la tarea. Luego, en una segunda etapa, se estiman los parámetros del modelo mediante un algoritmo de maximización de la esperanza EM modificado. En una tercera etapa (parte inferior de la figura), a partir del modelo y los nuevos parámetros de la tarea, se calcula una nueva trayectoria, usando una técnica de regresión de mezcla de gaussianas parametrizada (al que denominamos como PGMR [28]).

3.3.1. Estimación del Modelo con TPGMM

En la estimación del modelo, el cual se denominara TPGMM 1, las entradas son: un conjunto pequeño de demostraciones y los parámetros de la tarea. Estos últimos son marcos de referencia de puntos que guardan alguna relación con cada trayectoria demostrada. Por ejemplo, en la Figura **3-7**,

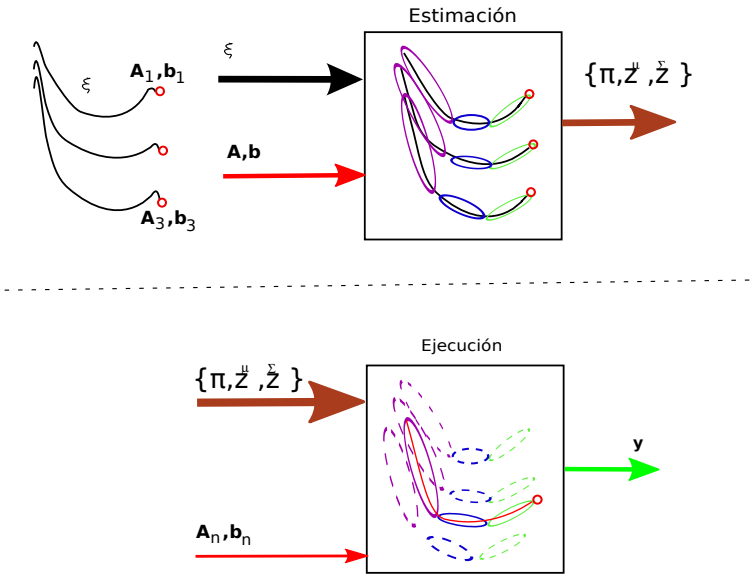


Figura 3-7: Diagrama de bloques de la técnica de modelo de mezcla de gaussianas parametrizado en la tarea. Parte superior: estimación del modelo. Inferior: cálculo de una nueva trayectoria.

los puntos de referencia son mostrados como círculos rojos al final de cada trayectoria demostrada. Igual que en GMM, previo a la aplicación del algoritmo EM, se obtienen unos parámetros iniciales del modelo a través del algoritmo de $k-medias$ o por división en el tiempo, el cual obtiene el vector de promedios y la matriz de covarianza al ubicar las gaussianas separadas en divisiones iguales en el tiempo. Una vez estimado el modelo con el algoritmo EM, se obtiene como salida la información compuesta por los coeficientes de mezcla, centros, y covarianzas de las gaussianas, que son los que denominamos parámetros del modelo.

Para la estimación del modelo se parte de M demostraciones, cada una con T_m puntos de datos, las cuales forman un conjunto de datos $\{\boldsymbol{\xi}_n\}_{n=1}^{N}$ con $N = \sum_{m=1}^{M} T_m$, cada $\boldsymbol{\xi}_n = [t_n, y_n]^T \in \mathbb{R}^{D+1}$ con D la dimensión de cada punto y t_n el instante de tiempo. Cada demostración se asocia con los parámetros de la tarea $\{\boldsymbol{A}_{n,j}, \boldsymbol{b}_{n,j}\}_{j=1}^{N_P}$ representando N_P marcos de referencia, con matrices de trasformación $\boldsymbol{A}_{n,j}$ y vectores de desplazamiento $\boldsymbol{b}_{n,j}$. Los índices n y j representan la muestra de tiempo n y el $j-\acute{e}simo$ marco de referencia.

Los parámetros del modelo son $\{\pi_i, \boldsymbol{Z}_{i,j}^{\mu}, \boldsymbol{Z}_{i,j}^{\Sigma}\}$, que representan los coeficientes de mezcla, los centros y las matrices de covarianza para cada marco j y la componente de mezcla i. Con estos parámetros del modelo, se calcula el centro resultante $\boldsymbol{\mu}_{n,i}$ y la matriz de covarianza $\boldsymbol{\Sigma}_{n,i}$ de cada

componente i, como el producto de las gaussianas linealmente transformadas:

$$\mathcal{N}(\boldsymbol{\mu}_{n,i}, \boldsymbol{\Sigma}_{n,i}) = \prod_{j=1}^{N_P} \mathcal{N}(\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\mu} + \boldsymbol{b}_{n,j}, \boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\Sigma}\boldsymbol{A}_{n,j}^{T}). \tag{3-36}$$

Usando la propiedad del producto de la distribución normal, el centroide y la matriz de covarianza de la ecuación anterior es:

$$\begin{aligned} \boldsymbol{\mu}_{n,i} &= \boldsymbol{\Sigma}_{n,i} \sum_{j=1}^{N_P} (\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\Sigma}\boldsymbol{A}_{n,j}^{T})^{-1}(\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\mu} + \boldsymbol{b}_{n,j}), \\ \boldsymbol{\Sigma}_{n,i} &= (\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\Sigma}\boldsymbol{A}_{n,j}^{T})^{-1}. \end{aligned} \tag{3-37}$$

Cada modelo GMM dependiente de un parámetro de la tarea, es un modelo local (independiente del modelo global), con el producto de los N_P modelos locales se obtiene un modelo global parametrizado de las trayectorias.

Los parámetros del modelo se estiman iterativamente usando una modificación del procedimiento EM [28]. Cada iteración consta de dos pasos, el primero calcula la probabilidad posterior y, el segundo, calcula los parámetros del modelo con base a esta probabilidad. La modificación con respecto al EM tradicional consiste en que el algoritmo toma en cuenta los parámetros de la tarea tal como se detalla a continuación.

3.3.2. Procedimiento EM Modificado

Los parámetros del modelo son estimados iterativamente usando una modificación del procedimiento EM (Maximización de la esperanza), propuesto en [28]. Cada iteración consta de dos pasos, el primero calcula la probabilidad posterior y el segundo calcula los parámetros del modelo con base en esta probabilidad. En el primer paso (E), la Ecuación 3-40 es usada con los parámetros temporales, para calcular la probabilidad. En el paso M, los parámetros del modelo son estimados a partir de esta probabilidad:

Paso E:

$$h_{n,i} = \frac{\pi_i \mathcal{N}(\boldsymbol{\xi}_n | \boldsymbol{\mu}_{n,i}, \boldsymbol{\Sigma}_{n,i})}{\sum_k^{N_K} \pi_k \mathcal{N}(\boldsymbol{\xi}_n | \boldsymbol{\mu}_{n,k}, \boldsymbol{\Sigma}_{n,k})}. \tag{3-38}$$

Paso M:

$$\begin{aligned} \pi_i &= \frac{\sum\limits_n h_{n,i}}{N}, \\ \boldsymbol{Z}_{i,j}^{\mu} &= \frac{\sum\limits_n h_{n,i}\boldsymbol{A}_{n,j}^{-1}[\boldsymbol{\xi}_n - \boldsymbol{b}_{n,j}]}{\sum\limits_n h_{n,i}}, \end{aligned} \tag{3-39}$$

$$\boldsymbol{Z}_{i,j}^{\Sigma} = \frac{\sum\limits_{n} h_{n,i} \boldsymbol{A}_{n,j}^{-1}[\boldsymbol{\xi}_n - \tilde{\boldsymbol{\mu}}_{n,i,j}][\boldsymbol{\xi}_n - \tilde{\boldsymbol{\mu}}_{n,i,j}]^T \boldsymbol{A}_{n,j}^{-T}}{\sum\limits_{n} h_{n,i}},$$

con: $\tilde{\boldsymbol{\mu}}_{n,i,j} = \boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\mu} + \boldsymbol{b}_{n,j}$.

3.3.3. Regresión de Mezcla de Gaussianas para el Modelo Parametrizado

La reproducción del modelo se realiza usado la técnica de regresión de mezcla de gaussianas (GMR). Esta no modela la función de regresión directamente, sino que modela una función de densidad de probabilidad conjunta de los datos y, a partir de ésta, deriva la función de regresión [19].

Esta técnica permite obtener la nueva trayectoria a partir del modelo de mezcla de gaussianas parametrizado en la tarea. Dados los parámetros del modelo $\{\pi_i, \boldsymbol{Z}_{i,j}^{\mu}, \boldsymbol{Z}_{i,j}^{\Sigma}\}$, los parámetros de la tarea $\{\boldsymbol{A}_{n,j}, \boldsymbol{b}_{n,j}\}_{j=1}^{N_P}$, se calculan en la iteración n, el centroide y la matriz de covarianza de la $i-$*ésima* gaussiana resultante como:

$$\begin{aligned} \boldsymbol{\mu}_{n,i} &\leftarrow \boldsymbol{\Sigma}_{n,i} \sum_{j=1}^{N_P} (\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\Sigma}\boldsymbol{A}_{n,j}^{T})^{-1}(\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\mu} + \boldsymbol{b}_{n,j}), \\ \boldsymbol{\Sigma}_{n,i} &\leftarrow (\boldsymbol{A}_{n,j}\boldsymbol{Z}_{i,j}^{\Sigma}\boldsymbol{A}_{n,j}^{T})^{-1}. \end{aligned} \tag{3-40}$$

Como cada trayectoria demostrada se compone del tiempo t, y las D variables de posición $\boldsymbol{y}$, las matrices se pueden descomponer de la siguiente manera: $\boldsymbol{\mu}_{n,i} = \begin{bmatrix} \mu_{n,i}^{t} \\ \mu_{n,i}^{\boldsymbol{y}} \end{bmatrix}$ y $\boldsymbol{\Sigma}_{n,i} = \begin{bmatrix} \Sigma_{n,i}^{t} & \Sigma_{i}^{t\boldsymbol{y}} \\ \Sigma_{n,i}^{\boldsymbol{y}t} & \Sigma_{n,i}^{\boldsymbol{y}} \end{bmatrix}$.
A partir de la regresión de mezcla de gaussianas (GMR), es posible recuperar el vector de salida y en un vector de tiempos t dado. Específicamente, GMR se basa en la distribución conjunta $\mathcal{P}(t, \boldsymbol{y})$ aprendida para el modelo PGMM parametrizado en la tarea, del cual se obtiene como la distribución de salida $\mathcal{N}(\hat{\boldsymbol{\mu}}_n^{\boldsymbol{y}}, \hat{\boldsymbol{\Sigma}}_n^{\boldsymbol{y}})$, que es también una gaussiana, con:

$$\hat{\boldsymbol{\mu}}_n^{\boldsymbol{y}} = \sum_{i=1}^{K} h_{n,i}(t_n)[\boldsymbol{\mu}_{n,i}^{\boldsymbol{y}} + \boldsymbol{\Sigma}_{n,i}^{\boldsymbol{y}t_n}(\Sigma_{n,i}^{t_n})^{-1}[t_n - \mu_{n,i}^{t_n}]], \tag{3-41}$$

$$\hat{\boldsymbol{y}}_n = \hat{\boldsymbol{\mu}}_n^{\boldsymbol{y}}, \tag{3-42}$$

la cual es la salida de la regresión y $h_{n,i}$ es:

$$h_{n,i}(t_n) = \frac{\pi_i \mathcal{N}(t_n | \mu_{n,i}^{t_n}, \Sigma_{n,i}^{t_n})}{\sum_{k}^{K} \pi_k \mathcal{N}(t_n | \mu_{n,k}^{t_n}, \Sigma_{n,k}^{t_n})}. \tag{3-43}$$

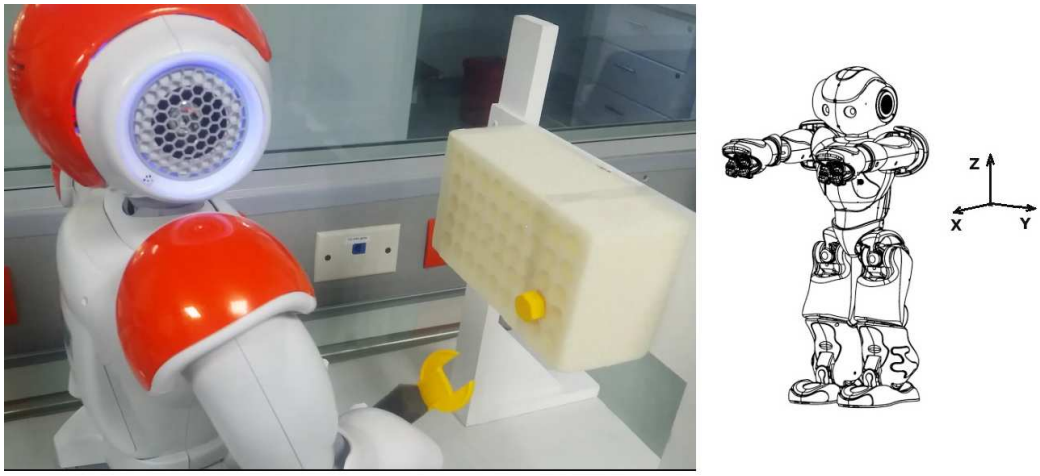


Figura 3-8: Configuración del experimento de llevar una llave a una tuerca, usando un robot NAO.

Variante de GMR con sistema dinámico (DS-GMR).
Considerando la trayectoria $\hat{\boldsymbol{y}}$ como un atractor y usando un controlador con constantes k^P, k^v, se calcula la salida $\boldsymbol{x}_n$:

$$\boldsymbol{x}_n = \boldsymbol{x}_n + \boldsymbol{v}_n T, \tag{3-44}$$

$$\boldsymbol{v}_n = \boldsymbol{v}_n + \boldsymbol{a}_n T, \tag{3-45}$$

$$\boldsymbol{a}_n = k^P[\hat{\boldsymbol{y}}_n - \boldsymbol{x}_n] - k^v \boldsymbol{v}_n, \tag{3-46}$$

donde $\boldsymbol{a}$, $\boldsymbol{v}$ y T son la aceleración, velocidad y tiempo de muestreo respectivamente.

Ejemplo 3.3

En la Figura **3-8** se muestra un robot NAO y el tablero utilizado [60] para la tarea de llevar una llave, a una tuerca usando programación por demostración. El bloque de espuma tiene diversos agujeros que permiten colocar el tornillo en varios puntos.

Se tomaron seis trayectorias demostradas de manera cinestática, lo que significa que se guía la mano del robot en una trayectoria, desde una posición de reposo a una posición final, que se logra cuando la mano ingresa la llave en la cabeza del tornillo. Además, a cada trayectoria se le realizó un recorte al principio y final, los puntos de recorte se seleccionaron de manera heurística, buscando evitar tener puntos donde las posiciones de la trayectoria varían ligeramente (ruido), pero que

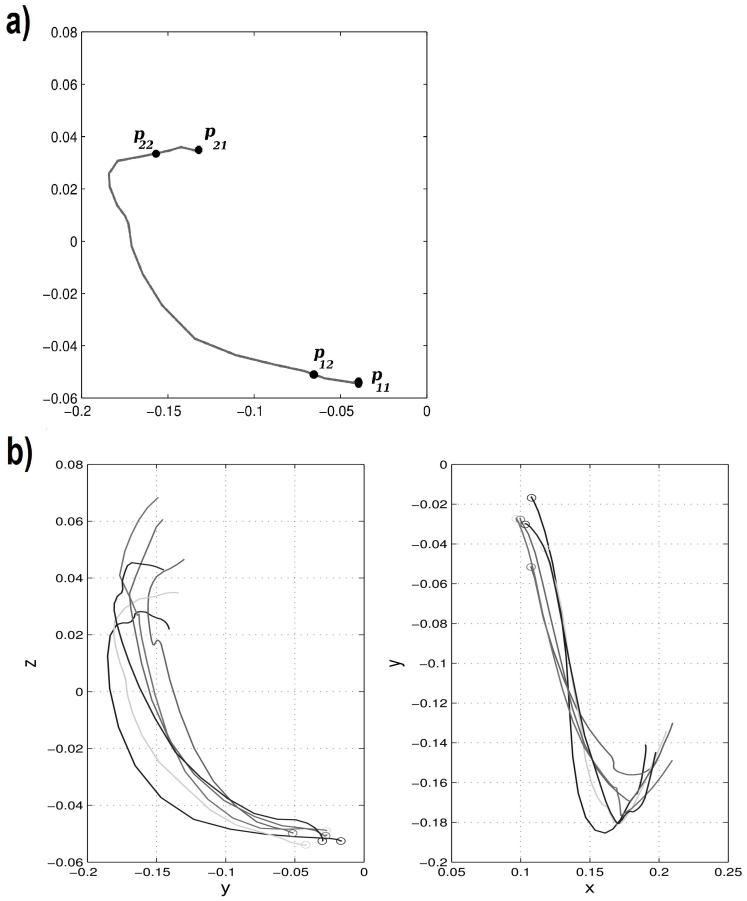


Figura 3-9: Trayectorias de las demostraciones capturadas con el mismo robot. Los círculos indican el inicio de la trayectoria. a) Obtención de los puntos p_1 y p_2. b) Trayectoria demostradas planos yz y xy.

pueden requerir gaussianas del modelo TPGMM, de manera innecesaria.

Para la estimación del modelo TPGMM, el número de gaussianas se tomó de manera heurística en seis, y se usaron dos parámetros de la tarea: a) posición inicial de la trayectoria demostrada y b) posición final. El primer parámetro, o dato inicial, se calcula a partir de los puntos p_1 y p_2. El punto p_{11} (Figura **3-9**a) es el dato inicial de cada trayectoria, y p_{12} se toma manualmente unas posiciones después de iniciar la trayectoria. Para el segundo parámetro o dato final, el punto p_{21}, se tomó como el final de cada trayectoria y el punto p_{22}, se seleccionó manualmente unas posiciones antes de terminar la trayectoria. En las figuras **3-9**b y **3-10**, se observan las demostraciones y las reproducciones de las mismas usando el modelo TPGMM. El marco de referencia de estas trayectorias es el mostrado en la Figura **3-10**.

Como además del modelo se requieren los parámetros de la tarea nuevos (puntos inicial y final), estos se obtuvieron así: El primer parámetro (punto inicial), p_{11} se definió como el promedio de

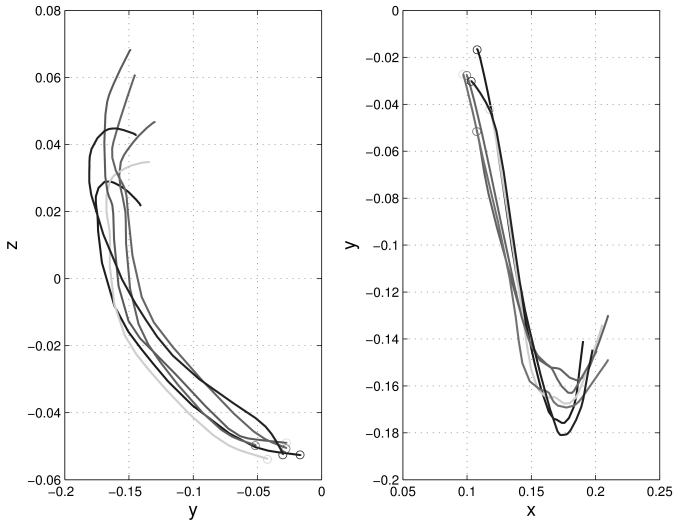


Figura 3-10: Trayectorias de las reproducciones usando los mismos parámetros de la tarea de las trayectorias demostradas. Los círculos indican el inicio de la trayectoria.

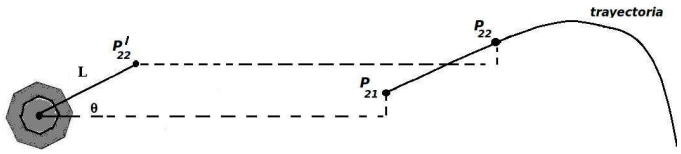


Figura 3-11: Cálculo de los puntos $p_{22}^{'}$ y p_{22}.

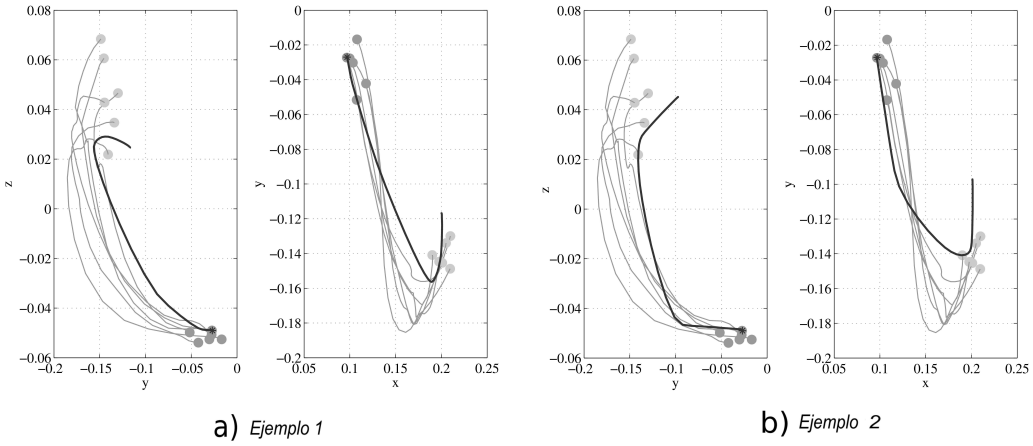


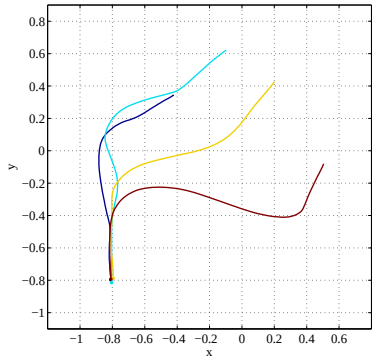
Figura 3-12: Ejemplos de generación de nuevas trayectorias. En gris las demostraciones, en negro, las trayectorias reproducidas para nuevos parámetros.

los datos de inicio de todas las trayectorias demostradas, y p_{12} se definió de manera similar al anterior. Para el segundo parámetro (punto final), p_{21} se obtuvo a partir de la posición del centro de la cabeza del tornillo (calculado por segmentación de imágenes), más un dato de traslación que lo lleva al marco de referencia de la mano del robot. El punto p_{22} se obtuvo a partir del cálculo del punto $p_{22}^{'}$ (Figura **3-11**), adicionando la misma translación comentada para p_{21}. El punto $p_{22}^{'}$, se obtiene como el punto a una distancia L, sobre la línea que parte del centro del tornillo y que tiene un ángulo de orientación θ de la cabeza del tornillo. En la Figura **3-12** se muestran dos ejemplos de trayectorias nuevas generadas (línea gruesa en color negro).

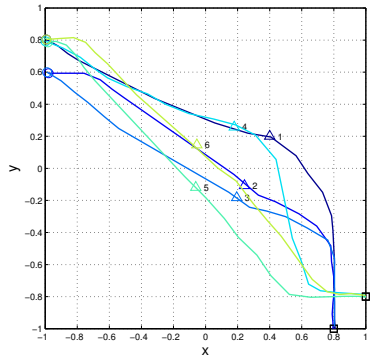
3.3.4. Criterio para la Cantidad de Gaussianas en un Modelo TPGMM

Aunque está definido el criterio de información Bayesiano (BIC) para GMM [24], no se lograron resultados satisfactorios al adaptar este a TPGMM, por lo que se optó por usar el error RMS de las M trayectorias demostradas usadas para estimar el modelo, como un criterio para la selección del número de gaussianas. En la Figura **3-13** se muestran dos conjuntos de trayectorias que, junto con el de las trayectorias del ejemplo 3.3 (Figura **3-9**), sirven para establecer un criterio para la selección del número mínimo de gaussianas. La primera tarea consiste en el parqueo de un carro desde diversas posiciones y las demostraciones fueron tomadas de Calinon y otros [28]. La segunda tarea consiste en la simulación de la tarea de barrer un objeto ubicado en una región central. La tercera tarea es la de llevar una llave a la cabeza de una tuerca con un robot NAO. A partir de las Figuras **3-14**a-c se plantea el siguiente criterio empírico: El número de gaussianas, es aquel en el cual se obtiene el primer mínimo del error RMS de las M trayectorias de la tarea.

En el caso de las demostraciones de parqueo es de 4 (Calinon y otros [28] emplearon 3) y barrer de 4 (Sección 6.3.1) y en el de llevar la llave es de 5. Al comparar los resultados con los valores seleccionados de manera heurística, estos son una unidad mayor (parqueo) o menor (llevar la llave). Al igual que el criterio BIC, la desventaja es que hay que evaluar el modelo para cada número de gaussianas, lo cual toma un tiempo considerable.

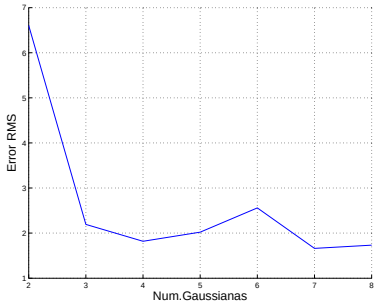


(a) Trayectorias de parqueo de un carro.

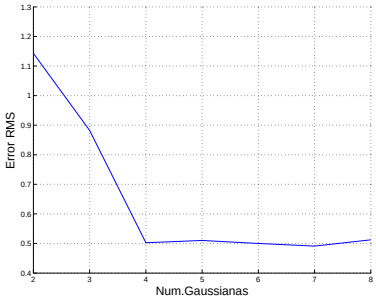


(b) Trayectorias de la tarea de barrer simulada.

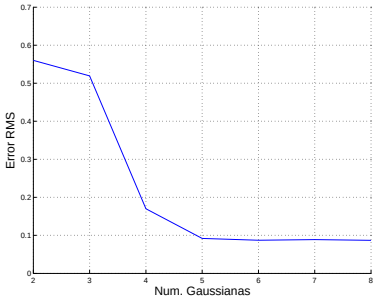
Figura 3-13: Dos de los tres conjuntos de trayectorias de tareas usados en la selección del número de gaussianas.



(a) Parqueo.



(b) Barrer.



(c) Llevar una llave a la cabeza de una tuerca.

Figura 3-14: Error RMS de las M trayectorias al variar el número de gaussianas.

3.4. Máquina de Aprendizaje Extremo (ELM)

Propuesta en el 2004 por Huang [61], es una red de una sola capa oculta en la cual las neuronas ocultas tiene una función de activación no lineal (Ej. función seno) y las neuronas de salida no tienen función de activación.

Para un conjunto de datos (X, Y) de n muestras, en el cual X es uno o d vectores de entrada y la variable Y tiene m vectores de salida, una red de una sola capa oculta se puede modelar como:

$$o = \sum_{i=1}^{l} \beta_i h(X/r, W_i, b_i), \tag{3-47}$$

con l neuronas ocultas, W los pesos que enlazan la capa de entrada a la capa oculta, b valores de desplazamiento, β la matriz de pesos que enlazan la capa oculta a la de salida, $h(\cdot)$ la función de activación de las neuronas de la capa oculta y r un factor de atenuación con un rango entre (2 a $n/10$), valores muy bajos o altos de r dan errores RMS grandes. Como se demostró en [61] y [62], la salida o de la red se puede aproximar a Y, siempre que se tenga una función h infinitamente diferenciable. Los pesos W se seleccionan aleatoriamente y los pesos β se calculan o ajustan con el conjunto de datos como se expondrá más adelante. El número de neuronas ocultas requeridas es menor o igual a la cantidad de datos $l \leq n$. Escribiendo la Ecuación (3-47) en forma compacta:

$$Y = \beta H, \tag{3-48}$$

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_n \end{bmatrix}_{n \times d}, \tag{3-49}$$

$$H = \begin{bmatrix} h(X_1, W_1, b_1) & \cdots & h(X_1, W_l, b_l) \\ \vdots & \cdots & \vdots \\ h(X_n, W_1, b_1) & \cdots & h(X_n, W_l, b_l) \end{bmatrix}_{n \times l}, \tag{3-50}$$

$$\beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_l \end{bmatrix}_{l \times m}, \tag{3-51}$$

$$Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}_{n \times m}. \tag{3-52}$$

El ajuste de los pesos β se realiza usando mínimos cuadrados, más específicamente, solucionando el sistema lineal $Y = \beta H$. Debido a que la matriz H es de tamaño $n \times l$ y a que es posible que esta tenga algunos valores singulares, generalmente se utiliza la operación seudo-inversa:

$$\beta = H^{\dagger}Y, \tag{3-53}$$

donde $H^{\dagger}$ es la matriz seudo-inversa, usando la ecuación de Moore-Penrose:

$$\beta = (H^T H)^{-1} H^T Y. \tag{3-54}$$

Como es demostrado en [62], esta solución tiene tres características:

- Mínimo error "global" de entrenamiento.
- Pesos β cuya norma es mínima.
- Solución única.

Uno de los problemas de la red ELM es que la matriz H puede llegar a ser singular, debido a la selección aleatoria de los pesos de entrada W y los desplazamientos b. Una solución a lo anterior es el uso de la regresión contraída [83] y [40]. Otra forma de producir una matriz H bien condicionada es la presentada por Hoang [59], donde calculan los desplazamientos a partir de las entradas y los pesos W aleatorios. Buscando la mejor solución para el problema de estabilidad, se implementaron dos algoritmos propuestos en [83] y [59]. Finalmente, después de algunas pruebas, se decidió usar el algoritmo de Hoang debido a que es más simple y, además, a que no incrementa el tiempo de aprendizaje. El cálculo de H empleando esta última técnica se resume en calcular los pesos W y los desplazamientos b, de la siguiente manera:

$$W = k_g randn(size), \tag{3-55}$$

donde k_g es un valor que se escoge de manera heurística y $randn$ genera números seudo-aleatorios de distribución normal.

Los valores de desplazamiento b, se pueden calcular como:

$$b = -diag(X^T W^T), \tag{3-56}$$

donde $diag$ significa la diagonal de la matriz del producto del vector de entradas X^T y los pesos W^T. A W se le podrían dar valores iniciales en lugar de aleatorios, por ejemplo es posible usar la diagonal de un cuadrado mágico si se tuviera un solo vector de entrada, obteniendo un buen resultado. Pero este cuadrado mágico, no funciona para cuando W es matriz, esto es porque los valores aleatorios son más diversos y garantizan generalmente mejores respuestas a la salida de la red.

Las ecuaciones (3-47) a (3-56) se implementaron en dos algoritmos, uno que realiza el entrenamiento de la red ELM (Algoritmo 1) y otro que hace el cómputo de la salida de la red (Algoritmo 2).

Algoritmo 1: Entrenamiento de la red ELM

Entrada: r,X,Y

Resultado: W, b, β

inicio

$X \leftarrow X/r$;

$W \leftarrow k_g randn(size)$;

$b \leftarrow -diag(X^T W^T)$;

$H \leftarrow \sin(XW + b)$;

$\beta \leftarrow H^{\dagger} Y$;

fin

Algoritmo 2: Cálculo de la salida de la red ELM

Entrada: r,W, b, β

Resultado: Y

inicio

$X \leftarrow X/r$;

$H \leftarrow \sin(XW + b)$;

$Y \leftarrow \beta H$;

fin

3.5. Resumen

Se presentaron dos extensiones del modelo de mezcla de gaussianas: técnicas incrementales y parametrización en la tarea. Esta técnica tiene la ventaja de modelar las variaciones de las demostraciones en términos estadísticos. Con el algoritmo de maximización de la esperanza, se obtienen los parámetros del modelo de múltiples trayectorias demostradas, luego la regresión de mezcla de gaussianas permite obtener una trayectoria que generaliza las múltiples trayectorias. El modelo también se puede obtener de manera incremental, adicionando nuevas trayectorias, en este aspecto se propusieron tres métodos: generativo, directo y de adicción de modelos. A través de ejemplos se mostró el funcionamiento de cada uno de ellos. El poder contar con parámetros adicionales que dependen de marcos de referencia en la tarea, permite poder generar nuevas trayectorias ante cambio en los parámetros, lo anterior, se ejemplificó con la tarea de llevar una llave a una tuerca usando un robot humanoide. Por último, se presentó la teoría de la red neuronal denominada máquina de aprendizaje extremo.

4 Máquina de Aprendizaje Extremo en Programación por Demostración

4.1. Introducción

Las redes neuronales fueron usadas en los inicios de la programación por demostración [66], [8], [13], [64], [68], pero debido a su lento ajuste y a que pueden caer en mínimos locales se les utilizó cada vez menos. En 2004 Huang [61] propuso una red neuronal a la que denominó máquina de aprendizaje extremo (ELM del inglés Extreme Learning Machine), la cual presenta un entrenamiento rápido y encuentra el mínimo global [62]. Básicamente, esta es una red neuronal de una sola capa oculta, en la cual no es necesario entrenar los pesos que van de la capa de entrada a la capa oculta. Además, los pesos que van de la capa oculta hacia la capa de salida se calculan en un solo paso, por lo que no se requiere un entrenamiento por iteraciones.

En este trabajo se emplea la máquina de aprendizaje extremo de dos formas: *i)* Para codificar la información de una o varias trayectorias de una misma demostración; y *ii)* Para aprender el modelo cinemático directo de un brazo de robot, el cual, una vez conocido, permite cambios en la posición final que alcanza la trayectoria demostrada. Una ventaja de aprender el modelo del brazo robot, es que el mismo programa se puede ejecutar en diferentes robots con modificaciones menores, por ejemplo cambiar el número de grados de libertad, lo cual es deseable en programación por demostración, donde se busca el mínimo esfuerzo en programación.

Se presenta como se generalizan varias trayectorias de una misma demostración, la comparación de la técnica de ELM con modelos ocultos de Markov continuos (HMM) [18], las primitivas dinámicas de movimiento (DMP) [133] y los modelos de mezcla gaussianas [21], a través de dos métricas (error RMS y tiempo de aprendizaje). La técnica es probada tanto en experimentos simulados, como en uno real. Las simulaciones se realizaron, en algunos casos, a partir de trayectorias reales tomadas con un guante de datos, estas son:

- Aprendizaje de las trayectorias de un robot de dos grados de libertad.
- Agarre con la mano de pelotas de diferentes tamaños.

En cuanto al experimento real, se utilizó el robot manipulador Katana 6M180 de cinco grados de libertad. Las pruebas que se realizaron con este robot fueron:

- Captura de trayectorias demostradas.
- Captura de datos con miras a la obtención del modelo cinemático con ELM.
- Pruebas ante cambios de la posición final.

Algunos trabajos similares son: reconocimiento de caracteres escritos de forma manual [30], reconocimiento de letras usando gestos manuales [124], reconocimiento de acciones humanas [94]. Sin embargo, estos trabajos no abordan el problema de modelado, la generalización de la demostraciones, ni la reproducción de los movimientos.

En la Sección 4.2 se presenta como se pueden codificar varias trayectorias de una misma demostración y como es posible alcanzar nuevas posiciones a partir de trayectorias demostradas. También se muestran algunas comparaciones con otras técnicas. En la Sección 4.3 se presentan algunas simulaciones y un experimento con un robot real, de los cuales se discuten los resultados y, finalmente, en la Sección 4.4 se presenta un resumen del capítulo.

4.2. Programación por Demostración y ELM

Además de modelar trayectorias, en PbD se requiere poder codificar múltiples demostraciones y también que se pueda variar la trayectoria modelada de las demostraciones para alcanzar ir a una nueva posición final, como por ejemplo cuando un objeto cambia de ubicación. A continuación se muestra como se resolvieron estos dos temas.

4.2.1. Codificación de Varias Demostraciones

Cuando se tienen n_d demostraciones de una misma tarea, la información de las señales de las demostraciones debe estar disponible en el entrenamiento, por lo que es necesario formar un conjunto de datos (X_a, Y_a) que contenga la información más relevante de las demostraciones. Por ejemplo para tres demostraciones este quedaría como:

$$X_a = \begin{bmatrix} X \\ -- \\ X \\ -- \\ X \end{bmatrix}_{3n} , \tag{4-1}$$

$$Y_a = \begin{bmatrix} Y_{dem1} \\ -- \\ Y_{dem2} \\ -- \\ Y_{dem3} \end{bmatrix}_{3n}, \tag{4-2}$$

donde X es el vector de tiempo discreto $[1, .., n]$ y $Y_{dem1}, Y_{dem2}, Y_{dem3}$, son las tres demostraciones. Reemplazando X por X_a en la Ecuación (3-49) y Y por Y_a en la Ecuación (3-52), se realiza el cómputo de β de acuerdo con la Ecuación 3-54. La red utilizada para codificar varias demostraciones, la denominaremos red ELM-1.

4.2.2. Cambio de la Posición Final

Generalmente, se requiere variar la trayectoria modelada de las demostraciones para que sea posible ir a una nueva posición final, como por ejemplo cuando un objeto cambia de ubicación. Calinon y otros [56] plantearon la mezcla de las salidas de un sistema dinámico con la trayectoria modelada usando regresión de mezcla gaussianas, el resultado de la mezcla produce señales similares a las modeladas pero que alcanzan un valor deseado de posición final, debido a que se agrega a las trayectorias originales el efecto del sistema dinámico. Como un cambio en las trayectorias en el espacio cartesiano implica cambios en las trayectorias de las articulaciones, Calinon utiliza una ecuación de optimización para calcular los nuevos valores de articulación [56], esta ecuación requiere el jacobiano del robot, lo que implica tener de antemano el modelo cinemático. Sin embargo, lo deseable en programación por demostración es que el mismo programa se ejecute en diferentes robots con pocas modificaciones. Aquí se plantea una variante al trabajo de Calinon, la cual en vez de utilizar una ecuación de optimización, realiza el cálculo de las articulaciones usando un algoritmo de cinemática inversa, basado en una red ELM que aprende el modelo cinemático directo del robot y, con el algoritmo de pseudo-inversa del jacobiano [7], calcula la cinemática inversa de cada punto en tiempo de ejecución.

A diferencia de Calinon y otros [56], que utilizan las ecuaciones del sistema dinámico, aquí se empleó la función descrita por la diferencia de las posiciones (4-3) y una función a tramos (4-4), donde V_{new} es la nueva posición del objeto y V_{dem} es el valor final de la trayectoria demostrada.

$$V_c = V_{new} - V_{dem}, \tag{4-3}$$

donde V_c es el nuevo valor que hace que la nueva trayectoria alcance V_{new}.

$$f(t) = \begin{cases} 0, & \text{para t entre 0 y 40 \%} \\ V_c t, & \text{para t entre 41 y 60 \%} \\ V_c, & \text{para t entre 61 y 99 \%} \end{cases}. \tag{4-4}$$

Los valores de división en el tiempo se seleccionaron de manera heurística y se optó por usar la función $f(t)$ con respecto a la ecuación dinámica empleada en [56], por considerarla más simple de implementar. Luego, sumando la salida del modelo que entrega la red ELM de la trayectoria $y_{ELM}(t)$, con la función descrita anteriormente, se obtiene la trayectoria que cambia al nuevo valor de posición:

$$y_{new}(t) = y_{ELM}(t) + f(t). \tag{4-5}$$

A partir de este valor se recalculan las nuevas trayectorias de articulación $q(t)$:

$$q(t) = ikine(y_{new}(t), q_0), \tag{4-6}$$

donde q_0 es un conjunto de valores de articulación inicial e $ikine$ es la función de cinemática inversa.

Aprendizaje de la Cinemática Directa del Robot Usando una Red ELM
En el cálculo numérico de la cinemática inversa por el método de seudo inversa del jacobiano [7], se requiere conocer el jacobiano, por lo que primero se presentará el algoritmo del jacobiano basado en la red ELM que aprende la cinemática directa y luego el algoritmo iterativo de cinemática inversa.

Algunos autores han planteado el uso de redes neuronales que evaden la necesidad de conocer el modelo cinemático del robot. Hasan y otros [55] plantean el uso de una red neuronal para obtener la cinemática inversa, esta es entrenada usando el algoritmo generalizado de retro-propagación, el cual es bastante lento comparado con la estimación de la red ELM. Feng y otros [136] emplean una combinación de un algoritmo de optimización con una red ELM, para aprender la cinemática inversa, la solución de esta cinemática inversa, es un problema mal condicionado, debido a que para una posición pueden existir múltiples articulaciones, por lo que los autores, previo al entrenamiento de la red, encuentran con el algoritmo de optimización, soluciones únicas a partir de una posición deseada y la posición actual, generando un subespacio que emplean para entrenar la red, los autores emplean la red ELM para aprender la cinemática inversa, a diferencia de este capítulo, donde se emplea la red ELM para encontrar la cinemática directa, y además se tiene en cuenta el problema de inestabilidad de esta clase de red, calculando los desplazamientos como en la Ecuación 3-56.

Los dos principales problemas de usar redes neuronales para aprender el modelo cinemático son obtener el conjunto de entrenamiento y el tiempo que se tarda en entrenar dicho conjunto. El primer problema es complejo de evadir, aunque una solución es reducir el conjunto de datos, acotando el espacio cartesiano donde el robot desempeñará sus tareas, en cuanto al segundo problema, con la red ELM se tiene la ventaja de no requerir iteraciones, por lo que el tiempo de entrenamiento se

reduce significativamente como se evidencia más adelante, en la sección de simulaciones y experimento.

Como la base del jacobiano es el cálculo del modelo cinemático directo, se entrenó una red ELM con n parejas de datos de posición y articulaciones del robot, la cantidad de neuronas l de la capa oculta se obtuvo gráficamente del error RMS contra el número de neuronas (Subsección 4.3.2), las entradas de la red son las articulaciones y las salidas son la posición del robot. El entrenamiento se realizó usando el Algoritmo 1, presentado en el capítulo 3, con lo que se obtienen los pesos de la red W_{kin}, b_{kin} y $Beta_{kin}$. A la red ELM que aprende la cinemática, en adelante se le denominara red ELM-2.

Para un robot de 6 grados de libertad, el jacobiano J está definido por:

$$J = \begin{bmatrix} \frac{\partial F_1}{\partial q_1} & \cdots & \frac{\partial F_1}{\partial q_6} \\ \vdots & \cdots & \vdots \\ \frac{\partial F_6}{\partial q_1} & \cdots & \frac{\partial F_6}{\partial q_6} \end{bmatrix}_{6\times 6}, \tag{4-7}$$

donde F_1 a F_6 son las ecuaciones de cinemática directa que describen el robot. Adaptando la matriz anterior en un algoritmo para nd grados de libertad y usando la red ELM como función de cinemática directa, se tiene el Algoritmo 3 con funciones de Matlab, que calcula el jacobiano.

Algoritmo 3: Cálculo del jacobiano con ELM (Jacob_ELM)

Entrada : $(\delta_q, q, W_{kin}, b_{kin}, Beta_{kin})$
Resultado: Je

$\delta_q \leftarrow 0{,}05$;
$Q \leftarrow [q_1 \ldots. q_{nd}]$;
$F \leftarrow ELM\_exec(Q, W_{kin}, b_{kin}, Beta_{kin})$;
for $i \leftarrow 1$ **to** nd **do**
  $\delta \leftarrow [zeros(1, i-1)\ \delta_q\ zeros(1, nd-i)]$;
  $Q1 \leftarrow Q + \delta$;
  $Fd \leftarrow ELM\_exec\ (Q1, W_{kin}, b_{kin}, Beta_{kin})$;
  $Je(:, i) \leftarrow Fd - F$;
$Je = Je/\delta_q$

En el algoritmo δ_q, es el valor de cambio en la variable de las articulaciones q del robot. El algoritmo primero calcula la cinemática directa para un valor dado del vector de articulaciones Q:

$$F = ELM\_exec(Q, W_{kin}, b_{kin}, Beta_{kin}). \tag{4-8}$$

Luego, para cada articulación, se calcula la cinemática directa pero incrementando δ para esta

articulación:

$$\frac{\partial F_i}{\partial q_i} \approx \frac{F([q1...q_i...q_{nd}] - F([q1...(\boldsymbol{q_i} + \boldsymbol{\delta})...q_{nd}]}{\delta}. \tag{4-9}$$

Estos valores se guardan en cada elemento de la matriz del jacobiano.

La Ecuación (4-10) se utiliza para comparar el jacobiano numérico basado en el modelo del robot, obtenido usando la librería de robótica para Matlab de Peter Corke [37], contra el obtenido usando el algoritmo 3. Esta ecuación consiste en la diferencia entre la suma del valor absoluto de los elementos del jacobiano con ELM y el de la librería de Corke, calculado con la siguiente ecuación:

$$error(k) = \sum_i \sum_j (|Jacob\_ELM(i,j)| - |Jacob(i,j)|), \tag{4-10}$$

para m puntos de prueba $\{q_1(k).....q_5(k)\}$; con $k = 1, ..., m$.

Para calcular la cinemática inversa, se utilizó el algoritmo de cálculo numérico usando la pseudo-inversa del jacobiano [7], que se muestra en el Algoritmo 4.

Algoritmo 4: Cinemática Inversa con ELM (CineInv_ELM)

Entrada : $(X, q0, W_{kin}, b_{kin}, Beta_{kin})$

Resultado: q

$\delta_q \leftarrow 0{,}05$;
$q \leftarrow q0$;
for $i \leftarrow 1$ **to** 500 **do**
  $Fq \leftarrow ELM\_exec(q, W_{kin}, b_{kin}, Beta_{kin})$;
  $dX \leftarrow X - Fq$;
  $J \leftarrow Jacob\_ELM(\delta_q, q, W_{kin}, b_{kin}, Beta_{kin})$;
  $dq \leftarrow pinv(J)dX$;
  $q \leftarrow q + dq$;

El Algoritmo 4 se fundamenta en la ecuación del jacobiano [7]:

$$J(q)_{i,j} = \left(\frac{\partial F_i(q)}{\partial q_j}\right)_{i,j}, \tag{4-11}$$

dado un dato cartesiano X y el resultado de la cinemática directa para un valor q (inicia con valor q_0), el algoritmo encuentra el vector q que minimiza el error dX:

$$dX = X - F(q), \tag{4-12}$$

y usando la pseudo-inversa y la Ecuación 4-11:

$$\Delta q = pinv(J)\delta X. \tag{4-13}$$

4.2.3. Cambio de la Posición Final por Estimación de la Trayectoria de Articulación

Un problema que puede presentar el algoritmo de cómputo de la cinemática inversa con ELM, es que la pseudo-inversa del jacobiano no existe en puntos de singularidad del robot [55]. Una opción, es calcular primero la trayectoria de articulación deseada θ^d y ejecutar esta trayectoria en el robot, obteniendo una trayectoria de posición deseada x^d. Pero esto no garantiza que esta trayectoria en el espacio cartesiano, guarde similitud con las trayectorias demostradas en el mismo espacio, por lo que Calinon y otros [16] proponen un método de optimización que encuentra la trayectoria cartesiana minimizando las diferencias entre las trayectorias de posición y articulación realizadas, con las de las trayectorias codificadas de las demostración, la función de costo es:

$$H(\dot{\theta}, \dot{x}) = \frac{1}{2}(\dot{\theta} - c)^T W_\theta (\dot{\theta} - c) + \frac{1}{2}(\dot{x} - d)^T W_x (\dot{x} - d), \tag{4-14}$$

donde $c = \theta - \theta^d$ y $d = x - x^d$, las trayectorias realizadas son (θ, x) , y las matrices W_θ y W_x son de ponderación, que hacen que las trayectorias se parezcan a las articulaciones o a las posiciones. La ecuación resultado de la optimización de la función de costo es:

$$\dot{\theta} = -(W_\theta + J^T W_x J)^{-1}(W_\theta c + J^T W_x d). \tag{4-15}$$

Con este valor se calcula la trayectoria de posición nueva, con el jacobiano:

$$\dot{x} = J\dot{\theta} \tag{4-16}$$

El procedimiento se resume en:

- Cálculo de los valores finales de articulación, a partir de la posición final deseada.
- Obtención de la trayectoria de articulación codificada, a partir de las demostraciones de articulación, usando ELM.
- Estimación de la trayectoria de articulación deseada, usando las ecuaciones (4-3), (4-4) y la trayectoria codificada.
- Cálculo de la trayectoria de posición deseada usando la cinemática directa.
- Se recalcula la trayectoria de articulación realizada, usando la Ecuación (4-15).
- A partir del jacobiano se recalcula la trayectoria de posición realizada x.

Si bien, el primer item del procedimiento, requiere el cálculo de la cinemática inversa, este se realiza para un solo punto y por una sola vez.

4.2.4. Evaluación de la Máquina de Aprendizaje Extremo

El desempeño de la red ELM se realizó a través de dos métricas, comparando los resultados de las métricas de la red ELM con los resultados de las métricas de otras tres técnicas, las métricas empleadas son el error RMS y el tiempo de estimación [20], tal como se presentó en el apartado 2.1.7. Las técnicas que se usaron para la comparación son:

- Modelo oculto de Markov continuo.
- Primitivas dinámicas de movimiento, versión correlacionada.
- Modelo de mezcla de gaussianas.

Para modelos ocultos de Markov (HMM), se utilizó la librería desarrollada para Matlab por Kevin Murphy [97], y se tomaron puntos claves de la trayectoria, como en Calinon en 2004 [18], para guardar en cada estado. Para primitivas dinámicas de movimiento (DMP), una versión desarrollada por Calinon y otros [22], en la cual es posible tener múltiples demostraciones de una misma tarea. En el caso de modelo de mezclas gaussianas (GMM/GMR), se utilizó la librería escrita por Calinon en 2009 [21]. Las anteriores librerías se tomaron de internet y se implementaron en un programa de Matlab, el cual se ejecutó en una CPU Intel Core Duo de 64 bits a 1,3 GHz corriendo en Windows 7.

Se realizaron comparaciones con dos conjuntos de datos, el primero es las trayectorias en línea punteada de la Figura **4-1**, el segundo es las trayectorias en el plano XY de letras. Para la comparación se varió la cantidad de elementos (estados: HMM; gaussianas: GMM y DMP; neuronas: ELM). Además, para cada cantidad se promedió el cómputo por 50 repeticiones. La red ELM se calculó con 100 muestras y el factor de atenuación r fue de 40 (este valor se aumentó al tener una salida de poca amplitud ($q_1, q_2 < 0{,}1$).

En la Figura **4-2**, se muestra el error RMS y el tiempo de estimación para el primer conjunto de datos, aunque no se muestran los resultados del segundo conjunto, la técnica presentó un tiempo de estimación del modelo un $40\,\%$ más bajo comparado con DMP con correlación (segundo más bajo), y en cuanto al error RMS, este es el segundo más bajo después de la técnica de GMM/GMR y cuyos valores la red ELM sigue muy de cerca. En cuanto a las varianzas del error RMS para ELM son menores a $0{,}25 \times 10^{-5}$ y las varianzas del tiempo de estimación son menores a $0{,}12 \times 10^{-3}$ que al sumarse a los valores, no traslapan con los errores RMS y los tiempos de las otras técnicas.

En las técnicas de DMP, HMM y GMM, se requiere un cómputo inicial que permite luego el entrenamiento de la técnica. En DMP se requiere el cálculo de la derivadas primera y segunda, en HMM y GMM se deben inicializar los centroides y anchos de las gaussianas. Este cómputo inicial no está adicionado en los tiempos de entrenamiento para HMM, DMP y GMM. Es de anotar que la máquina de aprendizaje extremo no requiere cómputo inicial.

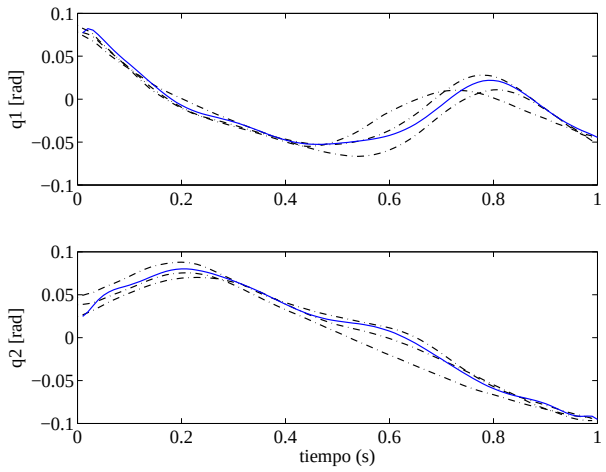
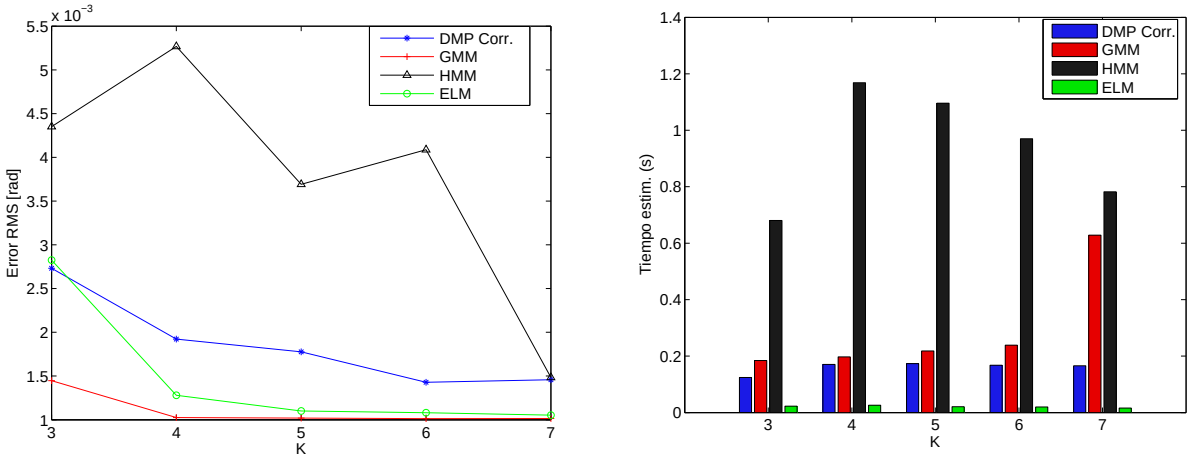


Figura 4-1: Respuesta de la red ELM para 7 neuronas ocultas, línea punteada: trayectoria demostraciones, azul: respuesta de la red ELM.



(a) Error RMS.

(b) Tiempo estimación.

Figura 4-2: Métricas al variar el número de elementos del modelo K.

4.3. Simulaciones y Experimento

4.3.1. Simulaciones

Como se mencionó, se realizaron varios experimentos para comprobar el buen funcionamiento de la máquina de aprendizaje extremo en aplicaciones de programación por demostración, estos se presentarán a continuación.

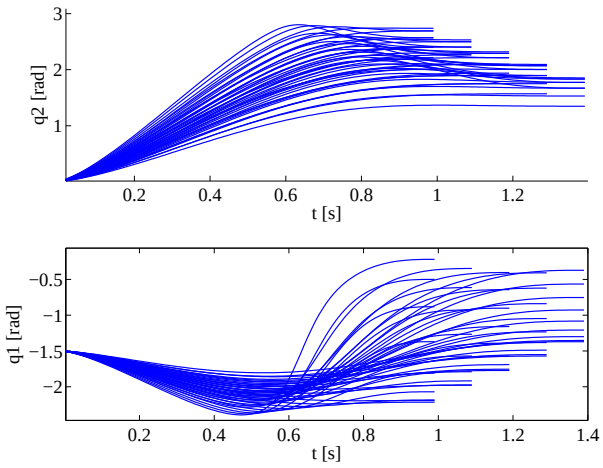


Figura 4-3: Trayectorias de las dos articulaciones al variar los 45 puntos x, y. Inferior: q_1, superior: articulación q_2.

Ajuste de Trayectorias de Jerk Mínimo de un Robot 2D.
Basado en el experimento simulado presentado por Ude y otros [133], se generaron 45 parejas de trayectorias usando la teoría de jerk mínimo [46], ubicando un robot de dos DOF en igual número de puntos del plano XY. En la Figura **4-3** se muestra las trayectorias de las dos articulaciones para los 45 puntos del plano. El robot tiene eslabones de $0{,}5$ metros y el rango de los puntos es de $-0{,}2$ a $0{,}6$ metros en x y de $-0{,}5$ a $0{,}3$ metros en y, con intervalos de $0{,}1$.

Se utilizaron dos redes ELM-1, cada una de 30 neuronas en la capa oculta, para ajustar las dos trayectorias de las articulaciones, cada una de las cuales tiene 140 muestras, el factor de atenuación r fue de 10. Debido a la naturaleza aleatoria al asignar los pesos de entrada W, el error RMS y el máximo varían. Esta variación se puede observar en una gráfica de error máximo contra número de repeticiones, el error máximo se calcula como en [133]:

$$e_{max} = \max_{k=1..T} \|y(k) - y_m(k)\|. \tag{4-17}$$

En la Figura **4-4** se muestra el error máximo para ambas articulaciones y para cada una de las 100 repeticiones, en las cuales se realiza un nuevo ajuste completo de la red ELM en cada repetición (asignación de valores de los pesos W y cálculo de β). El promedio y la desviación estándar del error fue de $0{,}43$ y $0{,}125$ grados respectivamente. En la Figura **4-4** se observa que el error máximo se mantiene menor a $0{,}011$ radianes ($0{,}7$ grados), el cual es aceptable para la tarea. El tiempo promedio de entrenamiento fue de $0{,}001$ s.

Ejemplo de Agarre de Pelotas.
Utilizando un guante de datos (Figura **4-5**), se recolectaron datos de las articulaciones de los dedos al agarrar tres pelotas de diferente tamaño, las trayectorias de las articulaciones se filtraron usan-

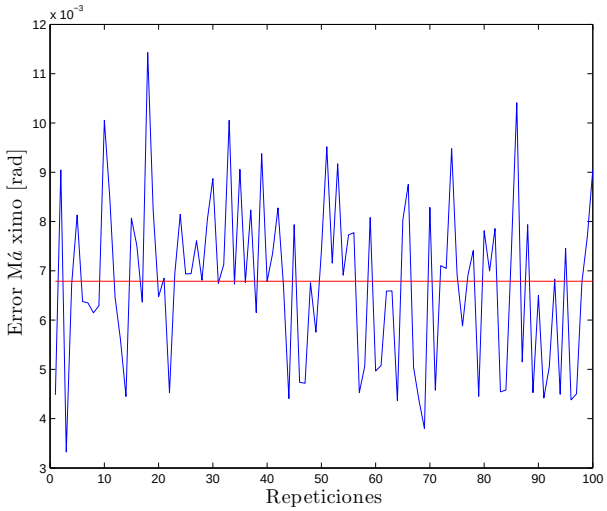


Figura 4-4: Error máximo para 100 repeticiones al ajustar con ELM, las 45 parejas de trayectorias de la Figura **4-3**.

Figura 4-5: Guante de datos.

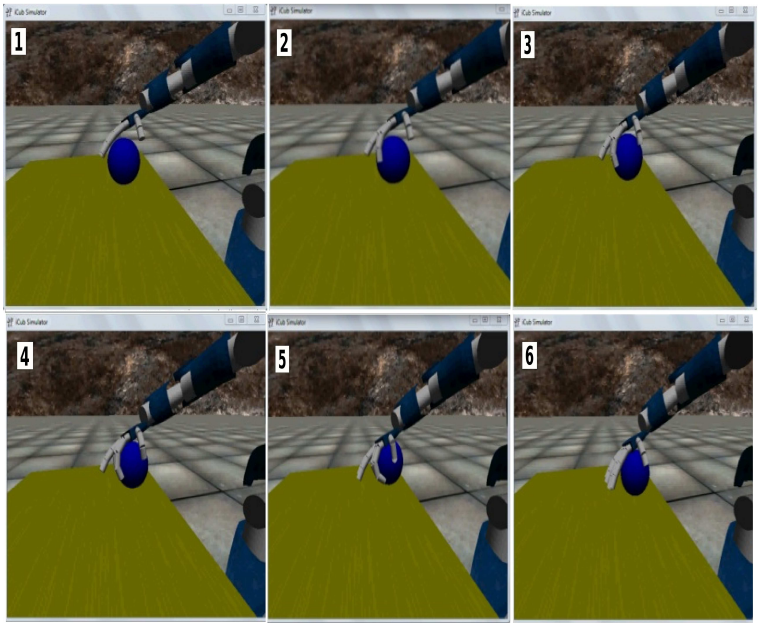


Figura 4-6: Secuencia del agarre de una pelota,se emplean los dedos (pulgar, índice y medio).

do un filtro de promedio móvil. Se usaron tres redes neuronales ELM-1 una para cada tamaño de pelota, las salidas de cada red son las trayectorias de las articulaciones de los dedos (pulgar, índice y medio), dos por dedo, en total seis trayectorias de 600 muestras cada una, el factor de atenuación r fue de 10. Cada red tiene 50 neuronas de entrada y 50×6 de salida.

En la Tabla **4-1** se muestra, para las trayectorias de agarre de los tres tamaños de pelota, los valores de las métricas después del entrenamiento con la red ELM. El error máximo no sobrepasó los $0{,}59$ grados y se redujo con respecto a la prueba de jerk mínimo.

Las trayectorias fueron probadas además en el simulador del robot Icub [131], en la Figura **4-6**, se muestra una secuencia del agarre de una pelota.

Discusión de los Resultados de las Simulaciones.
En los dos experimentos, el error máximo se mantuvo en un valor bajo. Por ejemplo, en la prueba con las 90 trayectorias de jerk mínimo y con las cuales se repitió el entrenamiento 100 veces, el error máximo se mantuvo menor a $0{,}8$ grados, lo que es una muestra experimental de la estabilidad. En la tarea de agarre de pelotas el error máximo no sobrepasó los $0{,}59$ grados y este se redujo con respecto a la prueba jerk mínimo. Se presentó un incremento en el tiempo de entrenamiento con respecto a la prueba de jerk mínimo, debido, principalmente, a que la cantidad de datos es trece veces mayor.

Tabla **4-1**: Valores de las métricas para la tarea de agarre pelotas.

Pelota	Métrica	Valor	Unidades
Pequeña	Error RMS		
	-Trayec. 1	0.016	
	-Trayec. 2	0.041	
	-Trayec. 3	0.024	
	-Trayec. 4	0.034	grados
	-Trayec. 5	0.026	
	-Trayec. 6	0.032	
	Error máximo (6 traject.)	0.59	
	Desviación estándar error RMS entre trayectorias	0.008	
	Tiempo de estimación	0.0185	s
Mediana	Error RMS		
	-Trayec. 1	0.026	
	-Trayec. 2	0.050	
	-Trayec. 3	0.030	
	-Trayec. 4	0.058	grados
	-Trayec. 5	0.025	
	-Trayec. 6	0.046	
	Error máximo	0.54	
	Desviación estándar error RMS entre trayectorias	0.014	
	Tiempo de estimación	0.0247	s
Grande	Error RMS		
	-Trajec. 1	0.028	
	-Trajec. 2	0.070	
	-Trajec. 3	0.026	
	-Trajec. 4	0.035	grados
	-Trajec. 5	0.027	
	-Trajec. 6	0.042	
	Error máximo	0.51	
	Desviación estándar error RMS entre trayectorias	0.017	
	Tiempo de estimación	0.0161	s

Figura 4-7: Robot Katana 6M180 usado para los experimentos.

4.3.2. Experimento Real

En la Figura **4-7** se muestra el robot Katana 6M180 de cinco grados de libertad, usado para las pruebas. Estas consisten en: Modelado de las trayectorias demostradas; cómputo del modelo cinemático con ELM; y cambio de la posición final.

Modelado de las Trayectorias Demostradas
La tarea consistió en tomar un objeto de una posición y ubicarlo en otra. Las trayectorias demostradas de esta tarea tanto en el espacio cartesiano como en el de las articulaciones se obtuvieron de manera cinestática.

Se usó una red ELM de 20 neuronas en la capa oculta para modelar las tres trayectorias cartesianas demostradas cada una de 100 valores. En la Figura **4-8**, se muestra en azul las trayectorias modeladas con la red ELM-1 uno y, en color negro y línea punteada, las trayectorias demostradas.

Cómputo del Modelo Cinemático con ELM.
Se obtuvo el modelo cinemático directo del robot usando una segunda red o red ELM-2. Para esto, se usaron 11340 vectores de articulación y se guardó la posición respectiva del efector final, leída directamente del controlador del robot. Los valores de articulación son tales que el espacio cartesiano resultante corresponde con el volumen donde el robot puede tomar o ubicar objetos. La reproducción por parte del robot de estas articulaciones y la obtención de las posiciones tardó aproximadamente dos horas.

Con las parejas de articulación y posición, se probó el desempeño de la red ELM-2, para aprender el modelo cinemático directo del robot. Para esta red se usaron 200 neuronas en la capa oculta (la selección de esta cantidad se presenta más adelante). El cómputo del algoritmo de entrenamiento tardó aproximadamente $2{,}9$ segundos. Una vez obtenidos los pesos, se probó la red en 36193 valores de articulaciones obtenidos usando un robot simulado con la librería de Corke [37], dentro

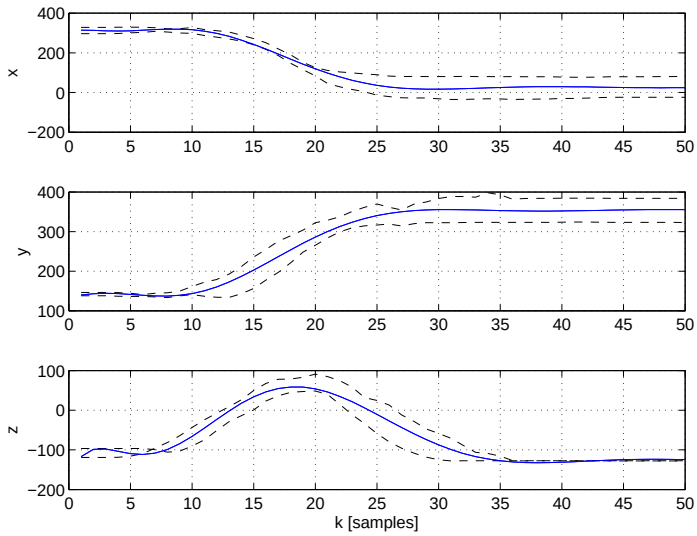


Figura 4-8: Trayectorias en el espacio cartesiano para tomar un objeto. Azul: trayectorias modeladas con la red ELM-1, línea negra punteada: Demostraciones.

del mismo volumen usado para el entrenamiento. Se usó un robot simulado, ya que esta cantidad de valores tomaría demasiado tiempo. En la Figura **4-9**, se muestra el histograma obtenido usando la Ecuación (4-10), el cual permite evidenciar que el error del jacobiano obtenido con la red ELM es bajo para los 36193 valores de articulaciones. Esta figura es discutida más adelante.

Número de Neuronas Contra Error RMS.
Para la red que aprende la cinemática directa, se muestra en la Figura **4-10** la variación de la cantidad de neuronas contra el error RMS para nuevos puntos o puntos no entrenados y el error RMS para puntos entrenados, este error es sobre los 11340 puntos, y los puntos nuevos son valores aleatorios de variación en un $20\,\%$ con respecto a los entrenados, nótese que para el error de nuevos puntos después de 250 neuronas, al incrementar el número de neuronas el error crece, esto es tal vez un problema de memorización.

Variación de la Posición Final.
Como es posible que existan varias demostraciones de la trayectoria sin cambio de posición y que además se desconozca la cinemática del robot, se emplearon dos redes neuronales: la primera codifica las demostraciones (ELM-1) y la segunda estima la cinemática directa (ELM-2). Para generar la nueva trayectoria al cambiar la posición final, se implementó el conjunto de ecuaciones (4-3) a (4-6), pero usando la cinemática inversa calculada con el Algoritmo 4, el cual usa la red ELM-2, cuyos pesos se obtuvieron en el experimento anterior. Los valores de la nueva posición del objeto V_{new} de la Ecuación (4-3), se tomaron con un par estéreo, usando cámaras web. Esta posición no se cambia una vez el robot está en movimiento. Una vez calculada la nueva trayectoria

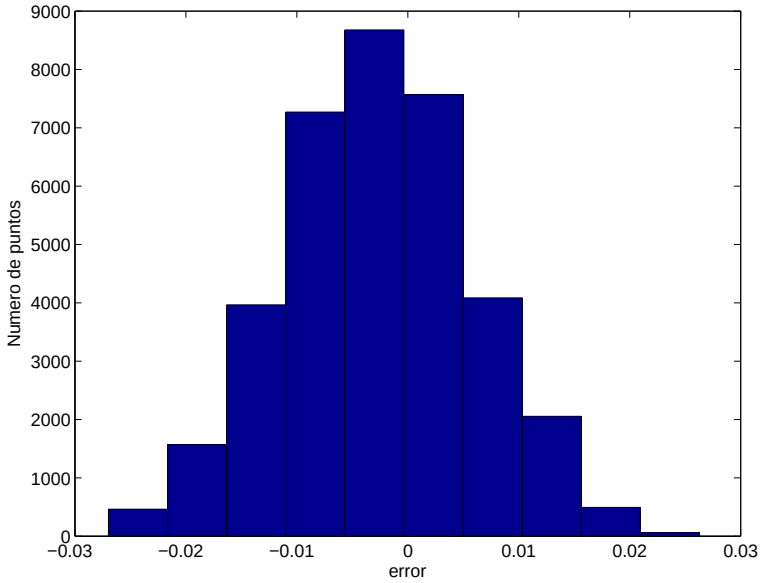


Figura 4-9: Histograma de la diferencia entre el jacobiano usando el modelo y usando la red ELM.

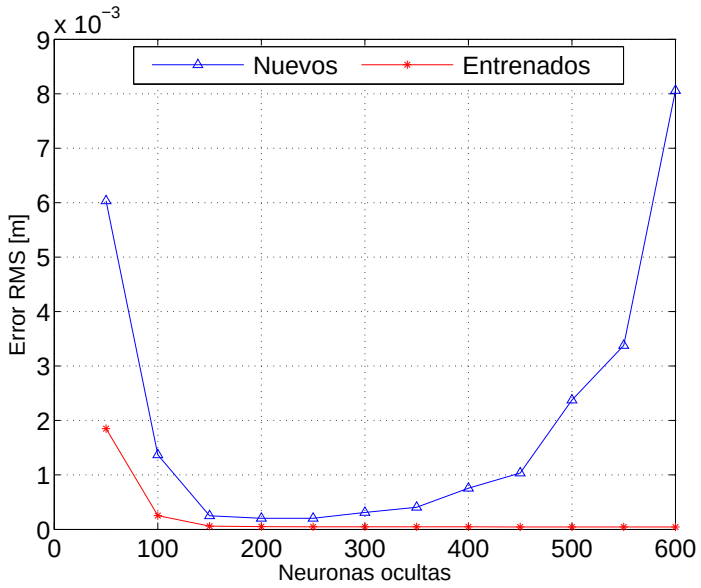


Figura 4-10: Error RMS para nuevos puntos y puntos entrenados contra número de neuronas.

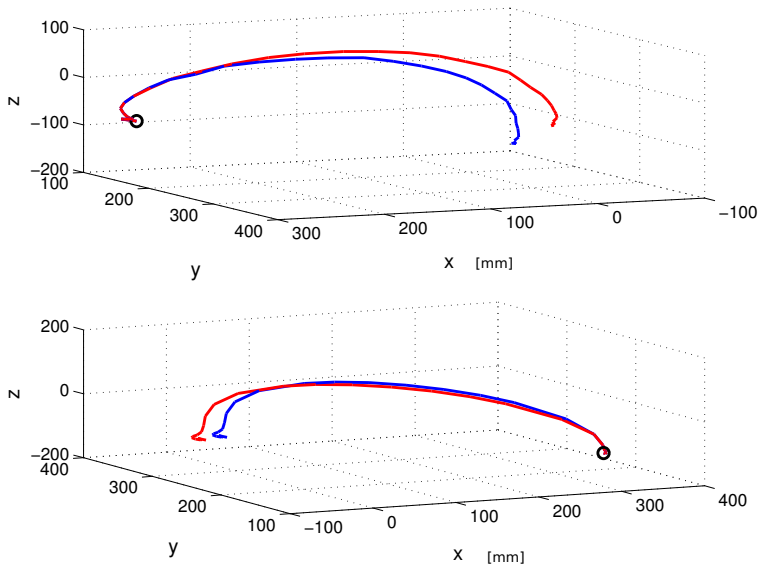


Figura 4-11: Dos ejemplos de trayectorias en el espacio cartesiano reproducidas por el robot al cambiar la posición final. Azul: Trayectoria modelada con la red ELM-1, Rojo: Trayectoria con cambio de la posición final. El círculo indica la posición inicial.

en el espacio de las articulaciones, esta es enviada al robot.

En la Figura **4-11**, se muestran dos ejemplos de trayectorias generadas al cambiar la posición final. En color azul, la trayectoria modelada con la red ELM obtenida a partir de las demostraciones y, en rojo, la nueva trayectoria similar a la demostrada, pero que alcanza el nuevo valor de posición. La información de estas trayectorias se tomó directamente del controlador del robot. En la Figura **4-11** al final del movimiento hay algunos cambios oscilatorios, estos son parte de la demostración inicial y no se removieron para mantener el movimiento lo más similar posible. Los cambios en la posición ocurren durante la trayectoria, dado el desplazamiento al agregar la respuesta de la función descrita por la Ecuación 4-4. Algunas imágenes de la ejecución de la trayectoria, se muestran en la Figura **4-12**. El primer cuadro es el inicio de la trayectoria, el final de la trayectoria es el cuadro número seis, es posible observar el movimiento curvilineo, equivalente al de la trayectoria en rojo mostrada en la Figura **4-11**.

Variación de la Posición Final por Estimación de la Trayectoria de Articulación.

Similar a la prueba anterior, se implementó el procedimiento descrito en la Sección 4.2.3, se usó el modelo cinemático directo del robot Katana, para el cómputo de la trayectoria de posición. Los valores de ponderación, empleados en la ecuación de optimización (4-15) que minimiza las diferencias entre las trayectorias de posición y articulación, son: $W_\theta = 0{,}5I_{4\times4}$, $W_x = 0{,}5I_{3\times3}$. En la Figura **4-13**, se muestran un gráfica de la trayectoria generada al cambiar la posición final a

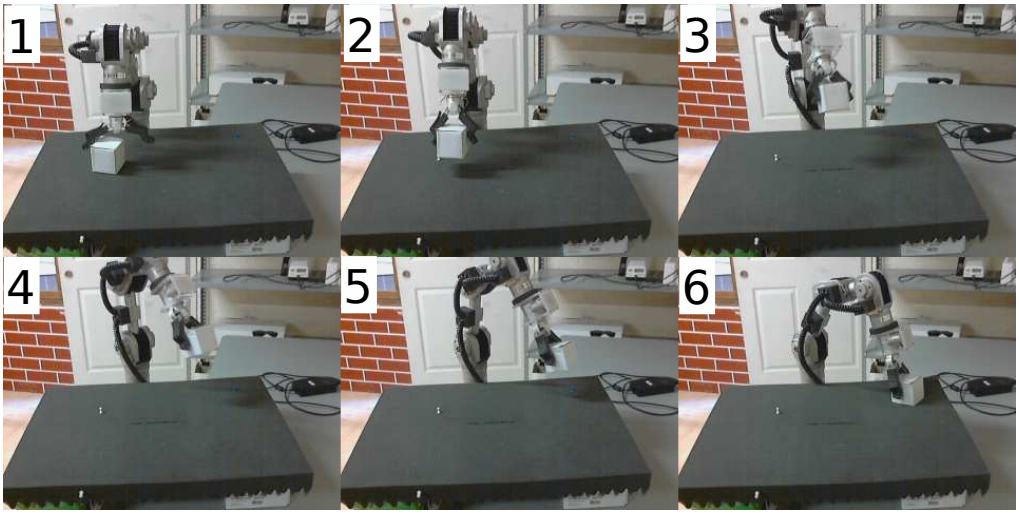


Figura 4-12: Imágenes de la ejecución de la trayectoria generada en rojo de la Figura **4-11**.

$(-20, 300, 140)$.

Discusión de los Resultados del Experimento Real.
Un efecto que ocurre al codificar con ELM es que cuando las trayectorias demostradas se mantienen constantes en el tiempo, la salida de la red puede presentar oscilaciones, como puede observarse al inicio de la respuesta de la variable z (Figura **4-8**).

Para aprender el modelo cinemático, en varias pruebas realizadas se evidenció que la red ELM requiere un alto número de neuronas (200 o más), esto debido a que se requieren una gran cantidad de parejas (posición, articulación) (11340 o más) para el entrenamiento, y esta cantidad depende del volumen de trabajo donde se desean generar las nuevas trayectorias. Sin embargo, el cómputo de los pesos se realiza de manera rápida (menos de tres segundos), y a partir del histograma de la Figura **4-9**, se puede ver que el error es bajo ($\pm 0{,}027$). Esto siempre y cuando se utilice un buen número de puntos, lo cual solo implica más tiempo para que el robot recorra estos puntos y con estos poder entrenar la red.

En otras pruebas con el robot Katana, este fue a diferentes posiciones finales, las pruebas permiten garantizar una apropiada trayectoria (no distorsionada y similar a la demostrada), siempre que no se exceda en un $30\,\%$ en el valor de la suma de los componentes x, y, z de la posición final de la trayectoria demostrada.

Generar la trayectoria empleando la técnica de cambio de la posición final por trayectorias de

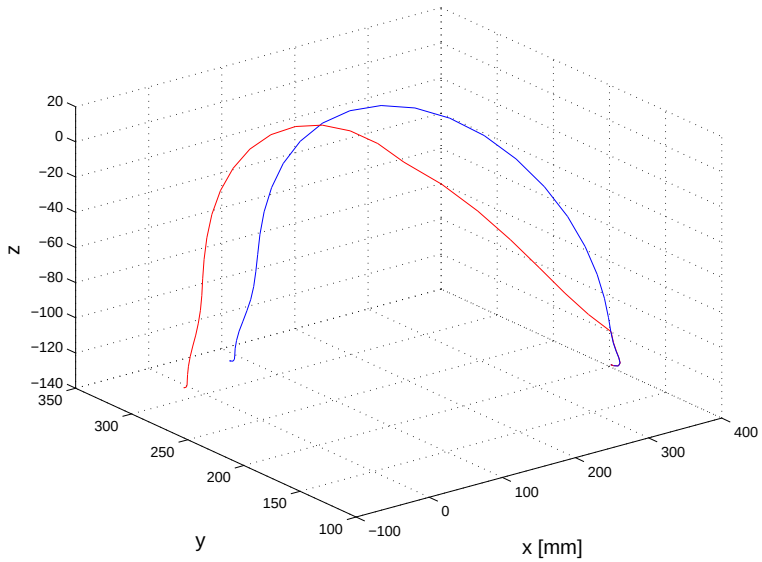


Figura 4-13: Ejemplo de trayectoria en el espacio cartesiano al cambiar la posición final. Azul: Trayectoria cartesiana original, estimada desde las articulaciones, Rojo: Trayectoria con cambio de la posición final, obtenida con estimación de la trayectoria de articulación.

articulación, produce resultados similares a los de emplear la técnica del cambio de posición por trayectorias cartesianas, con la ventaja de que se realiza de forma más rápida y se evade el problema de singularidades al calcular la cinemática inversa, la desventaja es que requiere las demostraciones en el espacio de articulación.

4.4. Resumen

Se presentó el desarrollo matemático que permite aplicar la red neuronal llamada máquina de aprendizaje extremo en PpD. Se presentan dos posibles aplicaciones de esta red, la primera permite la codificación de una o varias demostraciones de una tarea, la segunda permite el aprendizaje de la cinemática directa del robot a partir de la información de las articulaciones y la posición de efector final, con esta cinemática modelada y algunas ecuaciones, se generan nuevas trayectorias donde se cambia la ubicación final de la trayectoria. Mediante el uso de datos reales y simulación del robot, se cuantificó con métricas su comportamiento, también, a través de un experimento con un robot real, se analizó su comportamiento en la generalización de trayectorias demostradas.

5 Recuperación a Fallos de Ejecución en PpD de Robots Usando Múltiples Modelos

5.1. Introducción

La ejecución de una tarea por parte de un robot, en donde la trayectorias varían a causa de modificaciones en las posiciones de los objetos y, en la cual además, se manejan objetos deformables, puede presentar múltiples fallos de ejecución, lo que implica identificar diferentes momentos o estados previos al fallo y además reaccionar ante estas situaciones de tarea con diversas actuaciones. En este capítulo se propone una técnica que resuelve este problema, empleado un método de clasificación del fallo y múltiples modelos de mezcla de gaussianas [28], para realizar la recuperación del fallo de ejecución.

La tarea de poner una manga de camisa con un brazo robótico presenta una serie de retos interesantes. Por un lado, el robot debe operar cuidadosamente cerca de un humano, y la posición del brazo no es la misma. Por otro lado, cuando se pone la manga de una camisa se producen frecuentemente enganches de la camisa en el codo. Una persona cuando pone la camisa a un maniquí detecta el enganche y de forma natural retrocede, modifica un poco la posición, y retoma la trayectoria cambiándola ligeramente. Este es precisamente el desempeño que se quiere obtener del robot.

Con la técnica presentada en el Capítulo 4, es posible tener un parámetro de posición al final de la trayectoria, pero cuando la o las trayectorias demostradas dependen de varios parámetros que incluyen posición y orientación, es necesario emplear otras técnicas como los modelos de mezcla de gaussianas parametrizados en la tarea. Como el problema de recuperación a fallos implica múltiples parámetros, se empleó la última técnica mencionada.

En la aplicación de vestir mediante robots, existen pocos trabajos reportados hasta la fecha [120] y [129]. En [120] se presenta una aplicación en la que dos brazos robóticos colocan entre sus propios brazos una camiseta. Los autores usan una técnica que combina aprendizaje por refuerzo, visión estéreo, y coordenadas topológicas, para obtener un modelo de la tarea compuesto por el estado

del robot y el de la camiseta. Con un sistema de varias cámaras estéreo y marcas de color en las mangas, obtienen la información que les permite calcular un modelo topológico de estas. A partir del modelo, la técnica de aprendizaje por refuerzo genera las trayectorias de las articulaciones que ponen la camisa, aun ante cambios de la forma de la camiseta.

En [129], se presenta una trabajo similar al anterior, pero la tarea es colocar la camiseta sobre la cabeza de un maniquí. En este también usan aprendizaje por refuerzo y un modelo basado en coordenadas topológicas, tienen en cuenta para el modelo: *i)* la relación entre el cuello de la camisa y la cabeza del maniquí, *ii)* entre el cuello de la camisa y el cuerpo del maniquí y *iii)* entre la manga y el brazo derecho del maniquí. Con esta información y mediante aprendizaje por refuerzo el robot logra colocar la camiseta.

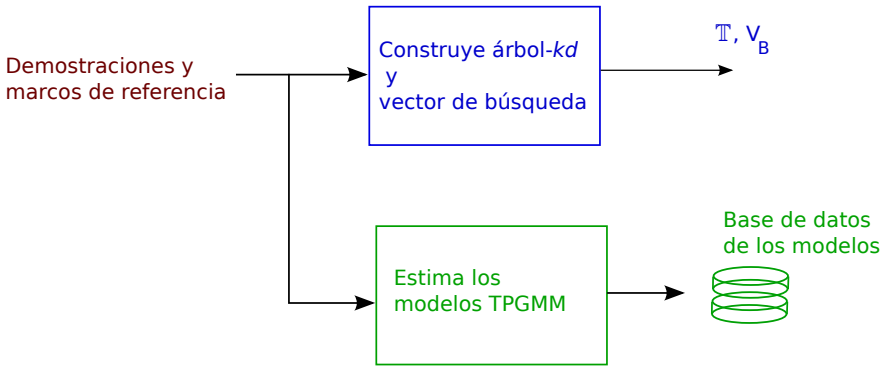
Nuestro trabajo se diferencia de los anteriores, en que estos usan aprendizaje por refuerzo y además requieren un modelo en coordenadas topológicas de las mangas y el hueco de la camiseta. También, con respecto a [129], el maniquí es fijo, a diferencia de nuestra propuesta que admite variaciones en posición y curvatura del brazo del maniquí. Por último, no toman en cuenta las fuerzas que se presentan en la tarea, para verificar el resultado de la colocación de la camiseta.

Los resultados del sistema completo se muestran a través de varios experimentos donde se coloca la manga de una camisa en el brazo de un maniquí. Algunas futuras aplicaciones posibles, son la ayuda a vestir a personas discapacitadas y ancianas. Aunque, al cambiar el maniquí por una persona puede simplificarse el problema (por autoayuda) o complicarse (por el movimiento).

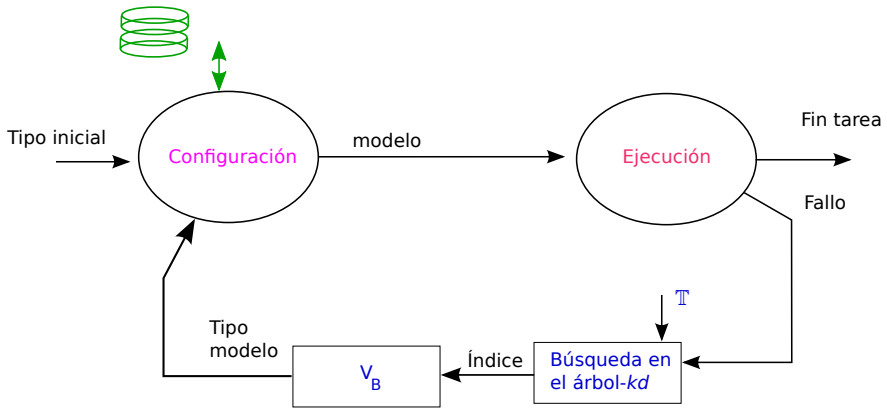
Lo que resta del capítulo presenta la siguiente estructura: en la Sección 5.2 se presenta cómo se estiman los modelos y el clasificador, también cómo es posible agregar una nueva recuperación a un fallo; en la Sección 5.3 se presenta la adquisición de las demostraciones del experimento realizado, posteriormente; en la Sección 5.4 se dan detalles de la estimación de la técnica para el experimento; en la Sección 5.5 algunos resultados y su discusión y; finalmente en la Sección 5.6 se presenta un resumen del capítulo.

5.2. Técnica Propuesta

La técnica propuesta consiste en un algoritmo que a partir del conjunto de demostraciones, con o sin fallo, aprende a recuperarse ante un fallo de ejecución (enganche). La técnica se compone de dos fases: *i)* Estimación y *ii)* Ejecución. Además, permite agregar nuevas respuesta a los fallos. En la Figura **5-1**, se muestran las dos fases del sistema propuesto.



(a) Fase de Estimación.



(b) Fase de Ejecución.

Figura 5-1: Diagramas de la técnica propuesta.

En la fase de estimación, con la información de todas las demostraciones, se realiza lo siguiente:

- Se estima un clasificador empleando un árbol-kd [11] al que se denominara $\mathbb{T}$, y que permite diferenciar una demostración de otra.
- Se segmenta de forma manual las demostraciones, según la tarea que cumplen (colocar la manga de camisa, recuperar un fallo).
- Con los segmentos se estiman diversos modelos de mezcla de gaussianas parametrizados en la tarea (Sección 3.3).
- Con los modelos obtenidos en el paso anterior se forma una base de datos de modelos.
- Se calcula un vector de búsqueda V_b, el cual convierte la salida del clasificador al tipo de modelo.

La base de datos (clasificador, modelos), luego puede ser ampliada, al re-estimar el árbol-kd y los modelos TPGMM, al presentarle demostraciones nuevas de como realizar la recuperación de un nuevo fallo.

En la fase de ejecución, dado un tipo inicial de modelo TPGMM, se generan los puntos de la trayectoria para la tarea normal (colocar la manga), al presentarse un fallo el árbol$-kd$ es consultado y, con esta información, se obtiene el nuevo tipo de modelo TPGMM a emplear, con el cual se recupera del fallo de ejecución.

5.2.1. Fase de Estimación

La estimación se realiza a partir de dos conjuntos de demostraciones: *i)* las que elaboran la tarea sin fallos de ejecución compuesto por M_{sf} demostraciones y *ii)* las de recuperación, que son trayectorias que permiten recuperar el fallo y terminar la tarea (M_{re} demostraciones).

Cada trayectoria demostrada y sus parámetros de la tarea son almacenadas en un objeto, al cual además se le agregaron dos etiquetas: *i)* Conjunto de pertenencia y *ii)* Fallo Ocasional. La primera, contiene una etiqueta según el conjunto de pertenencia $\{$Sin Fallo, Fallo 1$\}$, la segunda contiene información de la o las trayectorias que ocasionalmente presentan el fallo. La información de estas etiquetas fue grabada manualmente para cada objeto o trayectoria. Con la primera se genera de manera automática un vector de búsqueda V_b de tamaño $(M_{sf} + M_{re})$, en el cual cada elemento contiene el tipo de modelo usado.

Estimación del Árbol-*kd*.
Con la información de todas las demostraciones y sus respectivos parámetros de la tarea, se construye un árbol$-kd$ (en inglés kd-tree), el cual es una estructura que permite almacenar conjuntos de puntos (nodos) en un espacio $k-dimensional$ y es una generalización del árbol de búsqueda binaria.

Un nodo del árbol$-kd$, está formado por los N_P parámetros de la tarea (sin el dato de tiempo). Todos los nodos forman la matriz de datos $\boldsymbol{\eta}$, con la que se construye el árbol$-kd$ $\mathbb{T}$, dando como resultado una indexación de los nodos por similitud en sus valores.

$$\mathbb{T} = buid\_kdtree(\boldsymbol{\eta}), \tag{5-1}$$

donde $\boldsymbol{\eta}$ es de dimensión $(N_P \times D) \times (M_{sf} + M_{re})$.

Detección y Clasificación del Fallo.
La detección se realiza a través de una condición programada manualmente. La condición consiste en que la suma de las fuerzas en los tres ejes del sensor fuerza/torque, no sobrepase un valor umbral. Por otro lado, con la etiqueta de fallo ocasional, es posible saber si se trata de un fallo conocido (que ocurrió de manera previa para una trayectoria similar) y con el clasificador y el vector de búsqueda, es posible saber el modelo de recuperación.

5.2.2. Fase de Ejecución

El estado de configuración (Figura **5-1**), es el que define que modelo usar para cierta tarea (colocar manga, recuperación). A partir de un dato de parámetros de la tarea $\boldsymbol{P}_t$, se consulta al árbol$-kd$, para conocer cual es el nodo más cercano, obteniendo un índice i_x:

$$i_x = knn\_near(\boldsymbol{P}_t, \mathbb{T}). \tag{5-2}$$

Este índice y el vector de búsqueda V_b, permiten encontrar el tipo de modelo para recuperación de fallo en ese caso. Con este tipo, se extrae el modelo TPGMM de la base de datos. Inicialmente, se supone sin fallo, $\boldsymbol{P}_t$ se obtiene de los marcos de referencia para la tarea normal. En caso de fallo de ejecución, $\boldsymbol{P}_t$ es obtenida con los marcos de referencia, pero según el tipo de fallo. El sistema pasa a estado de ejecución, donde se reproduce la trayectoria y se chequea la condición de fallo. Si existe un fallo, se vuelve al estado de configuración.

En ciertos casos el dato $\boldsymbol{P}_t$ puede ser incorrecto, por lo que es necesario verificar la distancia Δ entre el nodo dado por el índice obtenido $\boldsymbol{P}_a$ y el dato suministrado:

$$\Delta = ||P_a(i_x) - \boldsymbol{P}_t||, \tag{5-3}$$

si esta distancia es mayor a $(0{,}12m)$, se considera que no fue encontrado un modelo solución de la tarea y se cae en el caso de modelo desconocido. Este último caso también se presenta si después de un fallo, la solución dada es el modelo sin fallo. El valor de distancia de $0{,}12m$ se obtuvo a partir de pruebas realizadas.

Resumiendo lo anterior, la técnica puede responder así:

- Ejecuta la tarea de manera normal.
- Se recupera de un fallo conocido.
- Informa y se detiene cuando cae en el caso de un fallo desconocido (no se encuentra una trayectoria similar donde ocurrió previamente).
- Informa y se detiene cuando cae en el caso de modelo desconocido (nodo alejado).

5.2.3. Inclusión de la Recuperación de un Nuevo Fallo

La técnica permite la inclusión de la recuperación de un nuevo fallo, para esto se le agrega la información de un conjunto de nuevas trayectorias que recuperan del fallo, lo cual se realiza en la fase de estimación, con los siguientes pasos:

- El usuario agrega la información de las demostraciones y marcos de referencia que permiten recuperar el nuevo fallo.
- El sistema incorpora la información en el árbol$-kd$ y estima además un modelo TPGMM, el cual se adiciona en la base de datos.
- El sistema vuelve a la fase de ejecución.

5.3. Adquisición de la Información

En la Figura **5-2**, se muestra una secuencia de imágenes de una demostración cinestática de la tarea de colocar la manga de la camisa. En este, una persona guía el efector final del robot con su mano, y se capturan los valores de articulación y pose del robot correspondientes a la trayectoria. Para la adquisición de los datos, se usó un robot WAM de siete grados de libertad y un brazo de maniquí. El efector final del robot tiene una pinza (Figura **5-3**) y un sensor de fuerza/torque (ATI-40). Con el sensor se adquieren las fuerzas en las tres coordenadas cartesianas, y los torques de los tres ángulos de orientación del efector final.

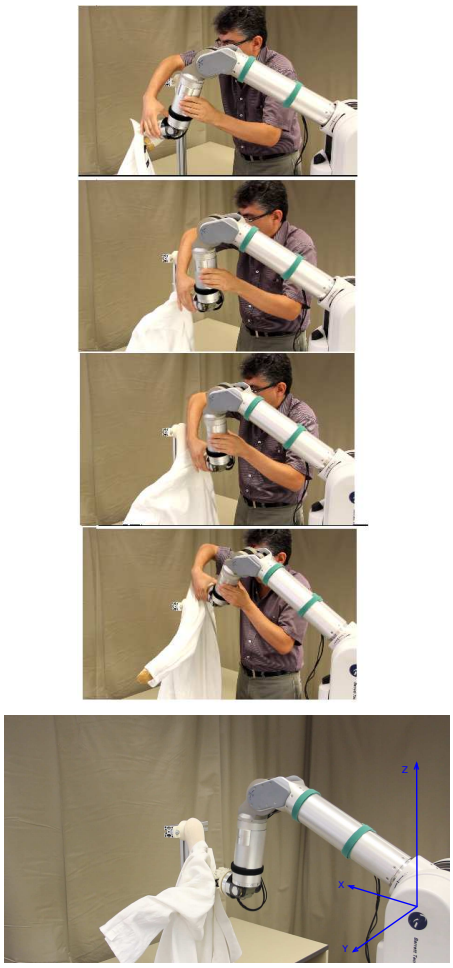


Figura 5-2: Demostración cinestática de una trayectoria. En color azul, el marco de referencia de la base del robot.

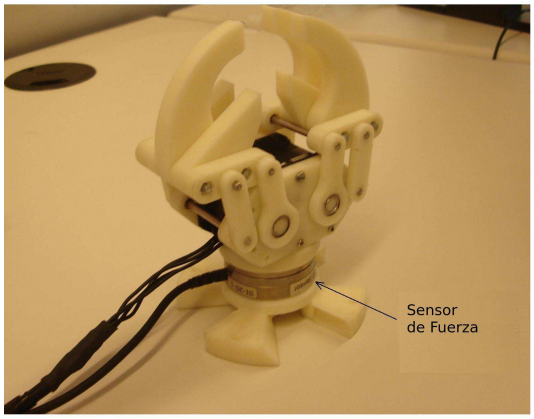


Figura 5-3: Pinza y el sensor de fuerza/torque.

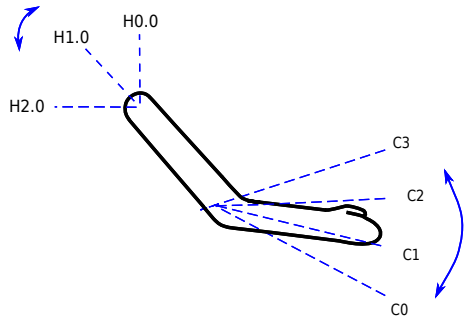


Figura 5-4: Diversas posiciones y curvaturas del brazo del maniquí.

El brazo del maniquí es de espuma forrada en tela, y permite variar la curvatura del codo del brazo y la posición espacial. Mediante una articulación en el hombro se puede girar de manera manual, para cambiar la posición hacia adelante o hacia atrás, como puede verse en la Figura **5-4**.

Este trabajo se centra en los aspectos de aprendizaje de la tarea y la estrategia de recuperación ante un fallo, por lo que la tarea de percepción se ha simplificado y los parámetros de la tarea se suministran manualmente al modelo, como se detalla en la Sección 5.4.

En el experimento se usaron cuatro posiciones, las cuales fueron marcadas en la articulación como *H0.0*, *H1.0*, *H1.5* y *H2.0* (Figura **5-4**), para hacer posible su posterior reproducción. El grado de curvatura del codo se varió entre curvatura tipo cero $C0$ (brazo extendido) y tipo tres $C3$ (aproximadamente 90 grados entre el antebrazo y el brazo).

Para cada demostración de la tarea de colocar la manga de la camisa, se configuró manualmente la posición y curvatura del brazo del maniquí. El agarre de la camisa por parte del robot se realizó manualmente, aunque puede ser automatizada [112]. Además de las trayectorias de demostración, se tomaron puntos en el brazo del maniquí, los cuales se usan para calcular los parámetros de la tarea. Toda la información y las trayectorias, está referenciada a la base del robot.

A. Trayectorias sin Fallo.
Al variar la posición de la mano y curvatura del brazo (Figura **5-4**), se puede inferir que se pueden producir diversas combinaciones. De estas, se seleccionaron de forma heurística seis, buscando que fueran muestras representativas, las cuales se listan en la Tabla **5-1**. Se aplicó una codificación con modelo de mezcla de gaussianas a tres demostraciones por cada uno de los seis items de la Tabla **5-1**. Las seis trayectorias resultantes de la codificación se muestran en la Figura **5-5**.

Tabla **5-1**: Combinaciones de posición de la mano y curvatura del codo que producen las trayectorias demostradas sin fallo.

Combinación	Descripción
H0.0 / C0	Brazo extendido con la mano en posición baja
H1.0 / C1	Brazo con curvatura de 120 grados con la mano en posición media
H0.0 / C3	Brazo cerrado a 90 grados con la mano en posición baja
H1.0 / C3	Brazo cerrado 90 grados con la mano en posición media
H1.5 / C3	Brazo cerrado 90 grados con la mano en posición media-alta
H2.0 / C3	Brazo cerrado 90 grados con la mano en posición más alta

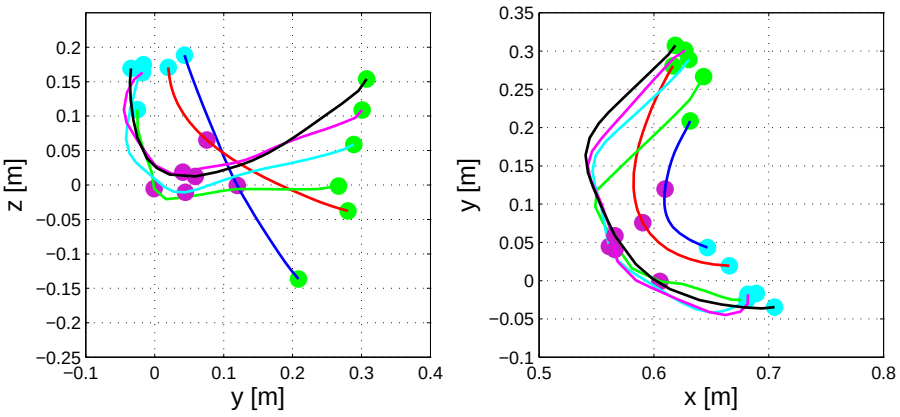


Figura 5-5: Trayectorias demostradas sin fallo, vistas desde los planos yz y xy. Los círculos en color (verde, violeta y azul), son los valores $\boldsymbol{b}$ en los parámetros de la tarea (mano, codo, hombro, respectivamente con cada color) para cada trayectoria.

B. Trayectorias de Recuperación.
En las Figuras **5-6** y **5-7**, se muestran cuatro tipos de fallos. El primer caso ocurre cuando la manga se engancha en el codo y no continua subiendo hasta el hombro, en la imagen de la izquierda el robot está colocando la manga, en la imagen de la derecha, se produce el enganche, y aunque el robot ha recorrido más la trayectoria de colocar la manga, la camisa continua en la misma posición que la imagen de la izquierda, pero se puede notar mucho más tensionada. En el segundo caso al momento de la inserción inicial, la mano del maniquí se choca con los bordes internos de la manga y la camisa se tensa. En el tercer caso (Figuras **5-7**), la manga no logra entrar y simplemente el robot pasa la manga por fuera del brazo. En el último caso, la manga se suelta de la garra. Dado que las fallas más frecuentes son la de enganche y la de inserción, estas son las discutidas a continuación.

En la Figura **5-8** se muestran las tres trayectorias de recuperación de un fallo de enganche de codo empleadas, en el plano yz y xy, codificadas con la técnica GMM a partir de dos demostraciones de cada una de las combinaciones de posiciones y curvatura tipo tres. En esta figura también se pueden ver los parámetros de codo y enganche, mostrados con círculos de color violeta y verde, respectivamente.

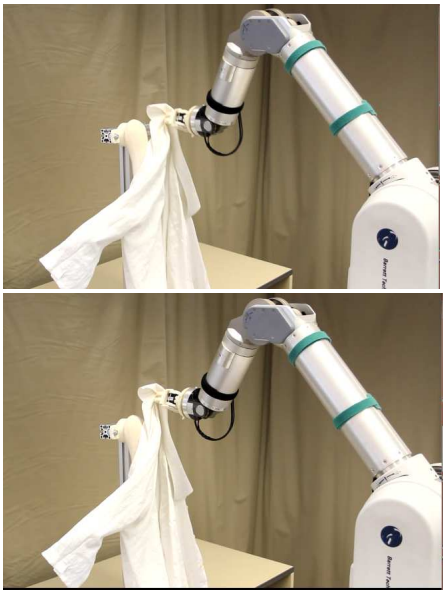
5.4. Detalles de la Fase de Estimación

Se presenta la estimación de los modelos, para lo cual primero, se presenta como se calcularon los parámetros de la tarea, y luego, se explica cómo se realiza la detección del fallo de ejecución.

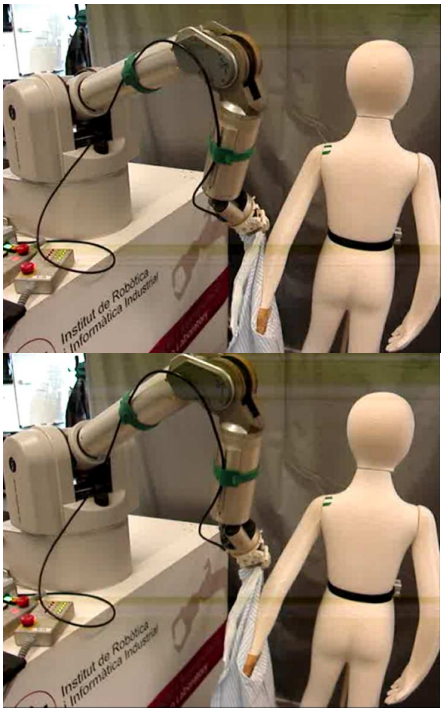
5.4.1. Cálculo de los Parámetros de la Tarea

Los parámetros de la tarea guardan la información sobre variaciones de posición y dirección de marcos de referencia, por ejemplo la mano o el hombro. Los parámetros de la tarea están compuestos por la matriz de transformación $\boldsymbol{A}$ (dirección) y el vector de desplazamiento $\boldsymbol{b}$ (posición). Se deben calcular u obtener tanto en el proceso de estimación del modelo, como cuando se pretende calcular una nueva trayectoria. El cálculo de $\boldsymbol{A}$ y $\boldsymbol{b}$, se realiza a partir de los puntos en el espacio cartesiano $\boldsymbol{p}_1$ y $\boldsymbol{p}_2$, referenciados a la base del robot WAM. Similar a [28] la matriz de transformación $\boldsymbol{A}$, se forma a partir de tres vectores $\boldsymbol{v}_1$,$\boldsymbol{v}_2$, y $\boldsymbol{v}_3$.

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & v_{1x} & v_{2x} & v_{3x} \\ 0 & v_{1y} & v_{2y} & v_{3y} \\ 0 & v_{1z} & v_{2z} & v_{3z} \end{bmatrix}. \tag{5-4}$$

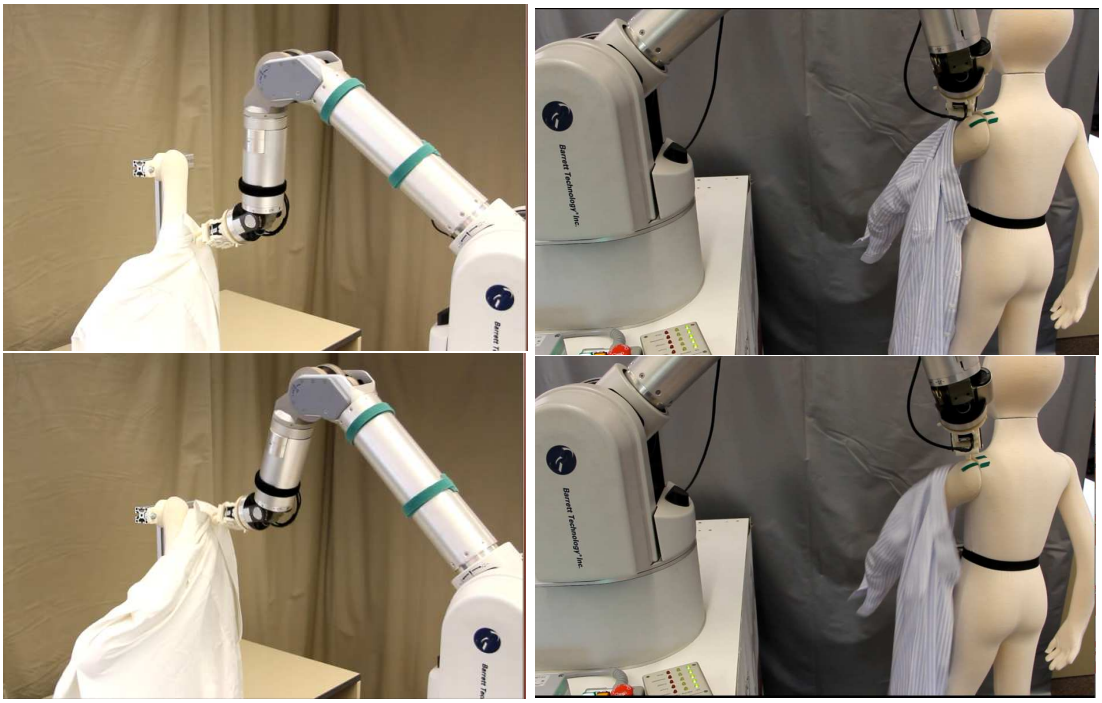


(a) Enganche de codo.



(b) Inserción inicial.

Figura 5-6: Imágenes de ejemplos de ocurrencia de fallos.



(a) Manga no entra. (b) Camisa se suelta de la garra.

Figura 5-7: Imágenes de ejemplos de ocurrencia de fallos (continuación).

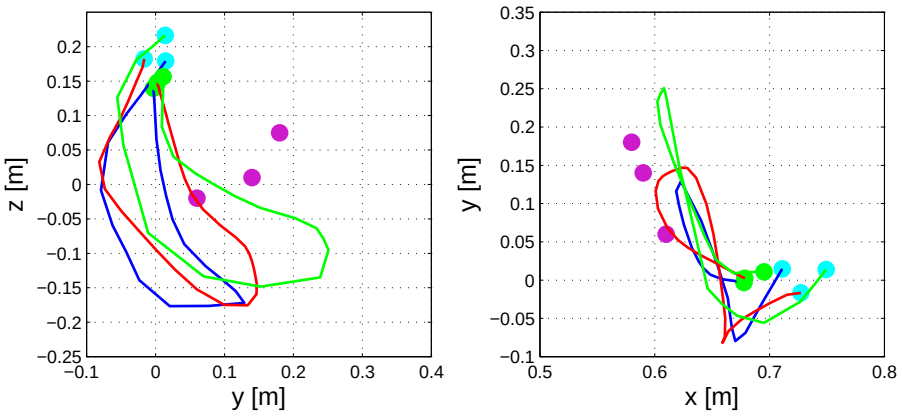


Figura 5-8: Trayectorias de recuperación de fallo de ejecución vistas desde los planos yz y xy. Los círculos en color (violeta y verde), son los valores de $\boldsymbol{b}$ en los parámetros de la tarea. En este caso el verde corresponde al punto de inicio de un fallo.

El vector $\boldsymbol{v}_2$ es el vector normal a la diferencia $\boldsymbol{p}_2 - \boldsymbol{p}_1$, y los vectores $\boldsymbol{v}_1$ y $\boldsymbol{v}_3$ son perpendiculares a $\boldsymbol{v}_2$, con lo que $\boldsymbol{A}$ es una matriz de transformación que contiene información de la orientación del punto $\boldsymbol{p}_1$ con respecto al punto $\boldsymbol{p}_2$. El vector de desplazamiento $\boldsymbol{b}$ se forma a partir del punto $\boldsymbol{p}_1$, como:

$$\boldsymbol{b} = [0, p_{1x}, p_{1y}, p_{1z}]^T. \tag{5-5}$$

5.4.2. Detalles del Modelo para las Trayectorias Sin Fallo

Para el modelo TPGMM de las trayectorias sin fallo, el número de gaussianas se seleccionó de manera heurística en seis, aunque también es posible aplicar el criterio presentado en la Subsección 3.3.4 para obtener este valor. Se usaron tres parámetros de la tarea: el primero es la posición inicial de la trayectoria, la cual básicamente está sobre la mano; el segundo es la posición del codo; y el tercero la posición del hombro. Estos se seleccionaron porque el primero guarda la variación de la posición donde se encuentra la mano, el segundo permite conservar la relación que existe entre la trayectoria y el punto de codo (curvatura), y el tercero suministra la posición del hombro.

Para el primer parámetro que corresponde a la mano, $\boldsymbol{p}_1$ es un punto cartesiano que está a dos centímetros sobre la mano, y $\boldsymbol{p}_2$ es el dato leído en el codo. En el caso del hombro, $\boldsymbol{p}_1$ es la posición del hombro y $\boldsymbol{p}_2$ es la del codo. Para los parámetros anteriores, el cálculo de $\boldsymbol{A}$ y $\boldsymbol{b}$ se realizó como se explicó anteriormente. Para el caso del codo, el vector de desplazamiento $\boldsymbol{b}$ se calculó con la posición del codo, y la matriz de transformación $\boldsymbol{A}$ se tomó como la matriz identidad de 4×4. En todos los casos, los puntos $\boldsymbol{p}_1$ y $\boldsymbol{p}_2$ se obtuvieron de manera manual, guiando el efector final a la posición deseada y almacenando dicha posición.

5.4.3. Detalles del Modelo de las Trayectorias de Recuperación del Fallo

En esta implementación se trabajó con curvaturas tipo tres, donde ocurren la mayoría de los enganches (Ver Figura **5-4**). En este caso, para estimar el modelo TPGMM, solo se usaron tres trayectorias correspondientes a cambios de posición del codo. Las trayectorias originales que reproducen la ruta de recuperación (regreso y avance), con las cuales se estima el modelo, se obtuvieron a partir de demostraciones cinestáticas y la captura de la información de los parámetros se realizó manualmente, el punto de codo se tomó colocando el efector final del robot en el codo y grabando su valor. Los marcos de referencia de puntos de enganche de las tres demostraciones se eligieron similares a los observados en pruebas preliminares, donde ocurrieron los enganches.

Se utilizaron seis estados o gaussianas y se seleccionaron tres parámetros de la tarea: el primero es un punto cercano al hombro (punto de enganche), el cual permite guardar la variación de punto de

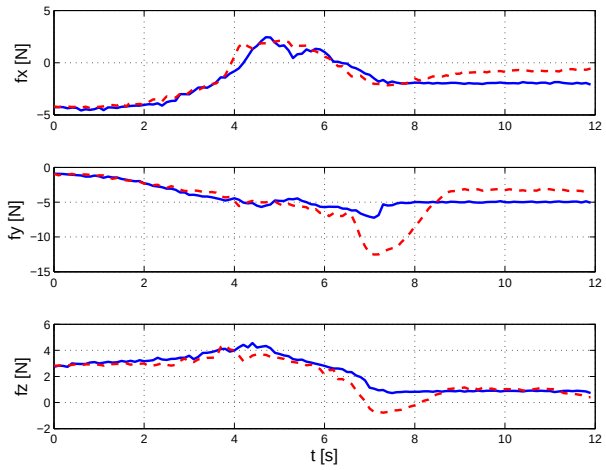


Figura 5-9: Dos casos de señales de fuerzas para la tarea de colocar la camisa. Azul/continua: La manga de la camisa logra ser colocada correctamente. En Rojo/punteada: Se presenta un enganche a los $7{,}3$ segundos.

inicio de la nueva trayectoria; el segundo es el parámetro de codo, que permite que la trayectoria cambie al variar la posición del codo; y el tercero la posición del hombro. En todos, la matriz de trasformación $\boldsymbol{A}$ se tomó por simplicidad como la identidad.

5.4.4. Detección del fallo de Ejecución

En la Figura **5-9** se presentan las fuerzas en el efector final del robot que permiten la detección del fallo. Las fuerzas en azul (línea continua), son las obtenidas durante una prueba donde la manga logra ser colocada correctamente. En rojo (línea punteada), se presenta un enganche, y se puede observar que la fuerza en el eje f_y se incrementa más de 10 Newtons. Experimentalmente se encontró un valor del umbral de fuerza de 12 Newtons, cuando se presentan enganches.

5.5. Simulación y Resultados

Se implementó un programa en Matlab de la técnica propuesta, se presentan a continuación resultados de la ejecución de los siguientes casos: trayectorias de casos base, simulación variando los parámetros de la tarea y simulación de la inclusión de una nueva recuperación.

5.5.1. Simulación de Ejecución de Trayectorias de Casos Base

Los casos base son aquellos en los que se emplean directamente, los valores de los parámetros de la tarea de las demostraciones usadas. En la Figura **5-10**, se muestra un ejemplo donde se simuló que la fuerza se mantiene constante todo el tiempo y menor a la fuerza de un enganche, por lo

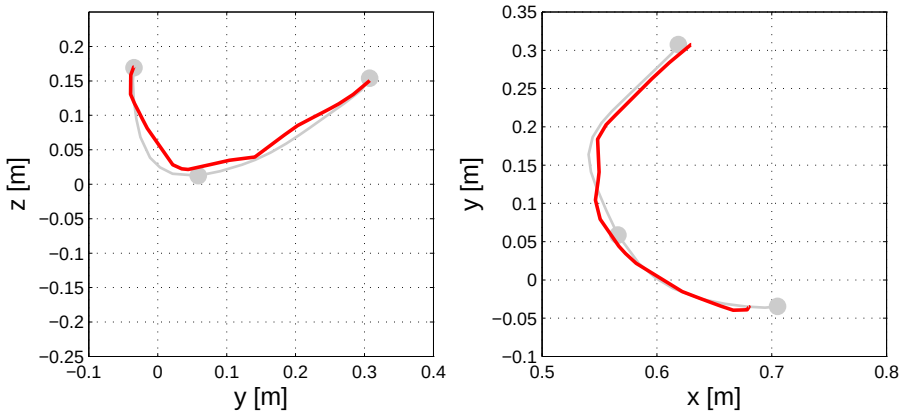


Figura 5-10: Ejemplo de simulación donde la fuerza permanece constante. Los parámetros de la tarea son de la trayectoria con curva $C3$ y posición $H2{,}0$.

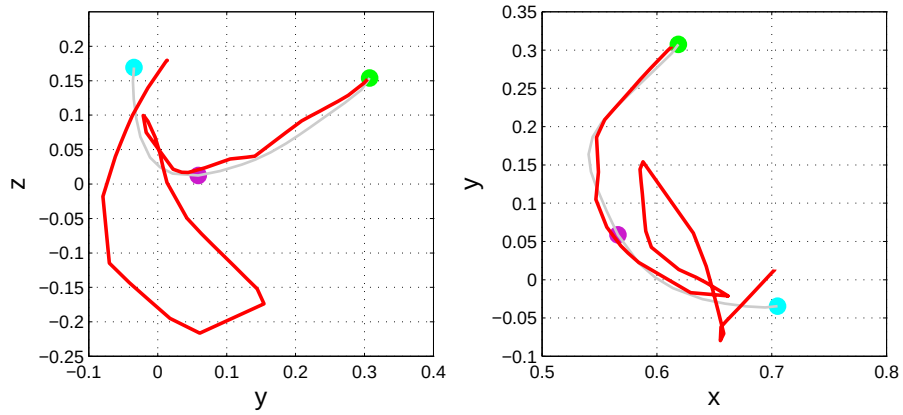


Figura 5-11: Ejemplo de simulación de la ocurrencia de fallo de ejecución. Los parámetros de la tarea son de la trayectoria con curva $C3$ y posición $H2{,}0$.

que el sistema no detecta fallo (No hay enganche). En rojo se muestra la trayectoria obtenida con GMR resultado del modelo TPGMM y en gris la trayectoria demostrada equivalente (curva C3 y posición H2.0). En la Figura **5-11**, se muestra la simulación donde la fuerza aumenta hasta el umbral de fallo en un tiempo $t = 7{,}3s$, el cual es el tiempo promedio donde ocurren los enganches cerca al hombro. En esta figura, se puede ver como la técnica produce una trayectoria (en rojo) que al ocurrir el enganche realiza la recuperación, la cual consiste en devolver la manga y regresar por una trayectoria diferente. Aunque en la Figura **5-11** se observan movimientos ligeramente bruscos, estos provienen de las trayectorias demostradas, como puede verse en la Figura **5-8**, estos son debido a que son dados por un humano. La posición final deseada que aparece como un punto azul claro, corresponde al de la demostración sin fallo y no al de recuperación, como el sistema ejecuta el modelo TPGMM de recuperación, se llega a la posición final de recuperación.

Se calculó, además, la diferencia punto a punto entre cada una de las seis trayectorias de entrenamiento (casos base sin fallo) y la simulación respectiva, dando un error RMS promedio de $1{,}2\,\%$ para los seis casos, y para los tres casos de recuperación del $2{,}2\,\%$.

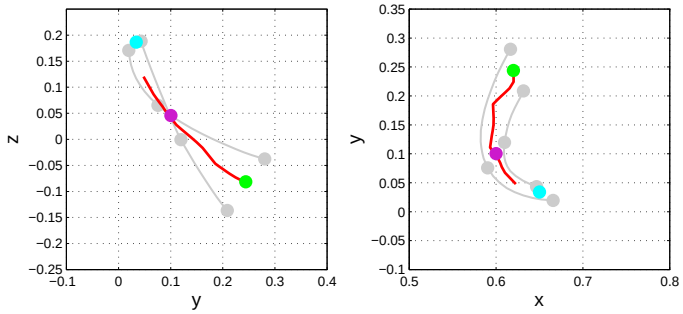
5.5.2. Simulación Variando los Parámetros

Se evaluó el uso de valores de los parámetros de la tarea diferentes a los empleados en la estimación del modelo. En la Figura **5-12**, se muestra varios casos de simulación (a-d). En estos casos, el aumento de fuerza hasta el valor de umbral se simuló en un tiempo $t = 7{,}3s$. En gris se muestran las trayectorias base más similares a estos nuevos casos, en rojo la respuesta de la técnica propuesta. Los casos **a-c** no producen trayectoria de recuperación, debido a que la técnica los considera como fallos desconocidos y detiene la ejecución. El caso **d** si produce trayectoria de recuperación, tal como se esperaba al usar la técnica propuesta.

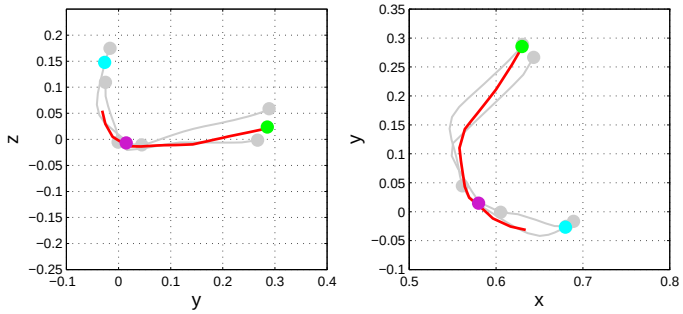
5.5.3. Simulación de la Inclusión de una Nueva Recuperación

En la Figura **5-13**, se muestran las cuatro trayectorias usadas para estimar un tercer modelo TPGMM, que permite recuperar ante fallos de ejecución al momento de inserción de la manga en el brazo, este fallo se produce porque en ocasiones el borde de la manga choca con la mano y genera fuerzas en el sensor. En este caso, la recuperación consiste en devolver la manga hasta un nuevo punto previo a la inserción de la manga en el brazo. Se utilizaron cuatro estados o gaussianas para el modelo TPGMM. En la Figura **5-14** se muestra un ejemplo de simulación de una recuperación. El fallo ocurre al inicio de la inserción de la manga. Las flechas indican los puntos de inicio del colocado de la manga (a) y ocurrencia del fallo de inserción (b).

En la Tabla **5-2**, se muestran casos de prueba realizados al incluir el nuevo fallo de ejecución, en cada caso se usan los parámetros de la tarea correspondiente a una trayectoria base o a los valores

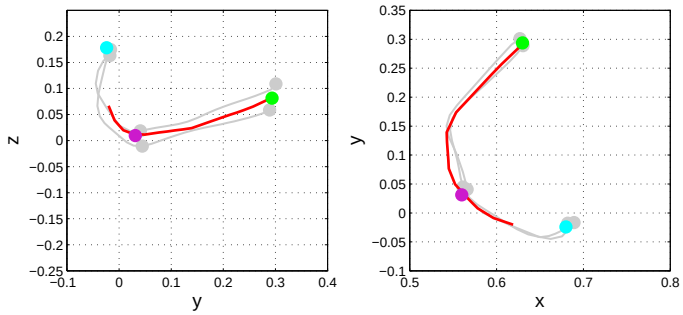


(a) Combinación de parámetros de las trayectorias 1 y 2.

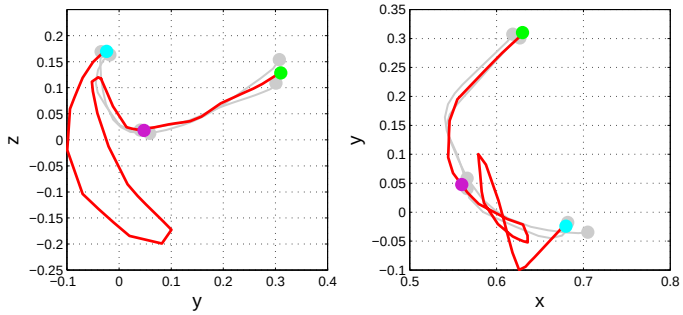


(b) Combinación de parámetros de las trayectorias 3 y 4.

Figura 5-12: Ejemplo de simulaciones de la ocurrencia de fallo de ejecución en $t = 7{,}3s$, para nuevos parámetros de la tarea. (Continúa en la siguiente página.)



(c) Combinación de parámetros de las trayectorias 4 y 5.



(d) Combinación de parámetros de las trayectorias 5 y 6.

Figura 5-12: Ejemplo de simulaciones de la ocurrencia de fallo de ejecución en $t = 7{,}3s$, para nuevos parámetros de la tarea (continuación).

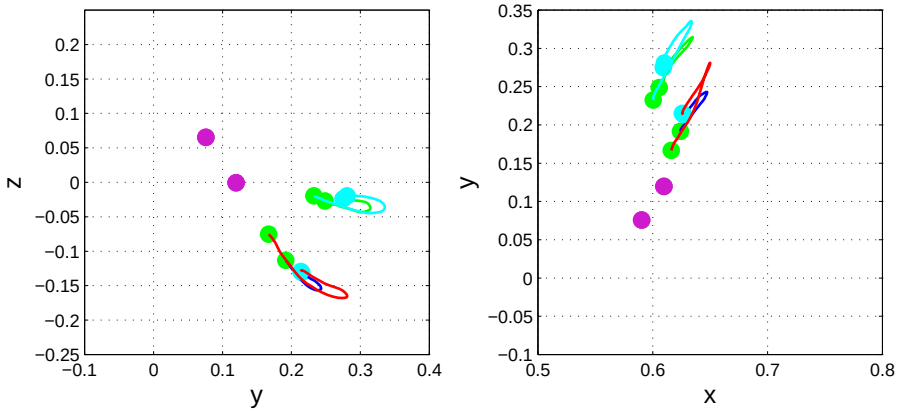


Figura 5-13: Trayectorias de recuperación de fallo de ejecución de inserción, vistas desde los planos yz y xy. Los círculos en color (verde, azul, violeta) son los valores de $\boldsymbol{b}$ en los parámetros de la tarea.

Tabla **5-2**: Pruebas realizadas y resumen resultados al variar los parámetros de la tarea y para dos casos de fallos.

Trayectoria o combinación	Fallo inserción	Fallo enganche
1	**R**	MD
2	**R**	MD
3	FD	FD
4	FD	FD
5	MD	**R**
6	MD	**R**
1 y 2	**R**	MD
3 y 4	FD	FD
5 y 6	MD	**R**

R: Recuperación, MD: Modelo desconocido, FD: Fallo Desconocido.

que resultan de combinar dos de ellas. En la primera columna, el número de trayectoria o los números de las trayectorias que se combinan para obtener los parámetros de la tarea del respectivo caso, en la segunda columna, las respuestas de la técnica ante un fallo de inserción que ocurre en $t = 3{,}2s$, en la tercera columna, las respuestas ante un enganche.

5.5.4. Discusión

Es de anotar, que en los casos **a-c**, de la Figura **5-12**, con aumento en la fuerza en el tiempo $t = 7{,}3s$, no se presentaron durante pruebas reales. Sin embargo, se optó por simularlos, para probar el comportamiento del algoritmo en diversas situaciones. La recuperación de la falla mostró que la acción correctiva fue adecuada y suficiente para el propósito de la tarea de colocar la manga.

Un problema que se puede presentar es que se seleccionen parámetros de la tarea para agregar un nuevo tipo de fallo de ejecución, que sean similares a los de uno existente en el árbol$-kd$. Esto ocasionaría una respuesta errónea como resultado de la consulta del árbol$-kd$.

5.6. Resumen

Se presentó una técnica que emplea múltiples modelos de mezcla de gaussianas parametrizados en la tarea, para resolver el problema de requerir diferentes comportamientos, ante diferentes fallos de ejecución de una tarea. A partir de demostraciones donde no ocurre fallo y demostraciones en que se le dice al robot como recuperarse de un fallo, se estiman múltiples modelos y se entrena

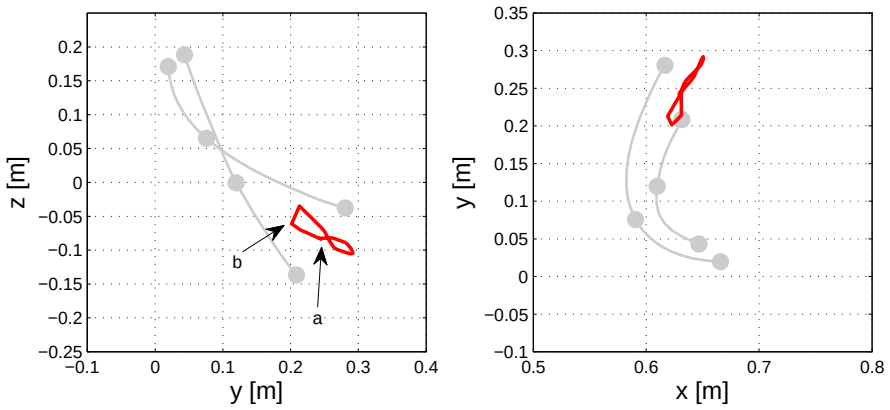


Figura 5-14: Ejemplo de simulación de la ocurrencia de fallo de ejecución. El fallo ocurre durante el inicio de la inserción de la manga. Las flechas indican los puntos de inicio del colocado de la manga (a) y ocurrencia del fallo de inserción (b).

un árbol$-kd$, el cual es usado como clasificador del tipo de fallo y permite conocer que modelo emplear para la recuperación. La técnica fue probada con la tarea de colocar la manga a un maniquí por un robot. El brazo del maniquí puede variar de posición y curvatura, y debido a que la manga de la camisa es un objeto deformable, esta puede tomar múltiples formas, con lo que se pueden presentar varios tipos de fallos de ejecución, en particular se analizaron dos: enganche de codo y enganche de inserción, y se mostró que la técnica los soluciona. La técnica permite fácilmente adicionar nuevos tipos de fallos.

6 Aprendizaje Incremental de un Modelo de Mezclas Parametrizado en la Tarea

6.1. Introducción

En este capítulo se presentan técnicas que permiten adicionar nuevas trayectorias a un modelo TPGMM existente, la técnica TPGMM es descrita en la Sección 3.3. Las trayectorias que se incrementan son diferentes a las usadas en la estimación del modelo TPGMM existente. Estas técnicas son: *i)* Generativa, *ii)* Adición de modelos y, *iii)* Actualización directa.

Las estrategias aquí presentadas suponen que no se conocen las trayectorias con que fue entrenado el modelo TPGMM existente, que el número de gaussianas es fijo tanto para el modelo existente como para el incrementado, y que dicho valor es adecuado para modelar suficientemente las trayectorias. La primera técnica consiste en generar trayectorias a partir del modelo existente, con estas y las que se desean incrementar, se forma un conjunto de datos que permite estimar un nuevo modelo. La segunda se basa en la técnica de adición de modelos de mezcla de gaussianas (GMM) propuesta en [54], esta permite adicionar los parámetros de los modelos directamente, en este caso se adicionan los parámetros del modelo existente a los parámetros del modelo estimado con las nuevas trayectorias. La tercera se fundamenta en una versión modificada de las ecuaciones del algoritmo de esperanza maximización, el cual estima los parámetros del modelo. Las ecuaciones separan los componentes correspondientes a los parámetros del modelo existente, de los componentes que requieren las trayectorias nuevas, con lo cual el modelo se actualiza directamente a partir de los parámetros existentes y las trayectorias nuevas.

La contribución consiste en que permiten obtener nuevos modelos TPGMM. Esto facilita el re-entrenamiento por demostración del robot, ya que no es necesario enseñarle nuevamente todas las trayectorias, o tenerlas almacenadas para su estimación. Aunque las técnicas generativa y de adición de modelos, si requieren, para este re-entrenamiento, almacenar los parámetros de la tarea de trayectorias previas, la técnica de actualización directa no los requieren.

El funcionamiento de las técnicas se comprobó a través de trayectorias simuladas y reales. En el primer caso, se utilizó un conjunto de trayectorias obtenidas manualmente con el ratón del computador. En el segundo caso se utilizaron trayectorias obtenidas con el robot WAM, ejecutando la tarea de colocar la manga de una camisa en el brazo de un maniquí, para diversos posiciones del brazo, tanto adelante o atrás. El cálculo del error RMS entre las demostraciones y las reproducciones, permite mostrar el desempeño de las técnicas y como las técnicas mejoran la respuesta del modelo con respecto al caso donde solo se usa el modelo existente.

En la Sección 6.2 se presentan las estrategias propuestas, posteriormente, en la Sección 6.3, se presenta las pruebas, sus resultados y discusión, finalmente se presentan las conclusiones.

6.2. Técnicas para Incrementar Trayectorias

El cálculo de las técnicas propuestas requiere conocer la información existente y la de las nuevas trayectorias que se desean agregar. La información existente consta, de los parámetros del modelo TPGMM (Sección 3.3), y de los parámetros de la tarea del conjunto de trayectorias originales. La información de las nuevas trayectorias se compone de las posiciones y orientaciones que describen cada trayectoria, y de los parámetros de la tarea correspondientes con la trayectoria.

La técnica generativa y la técnica de actualización directa están basadas en [27] y la técnica de adición de modelos GMM en Hall y Hikcs [54]. El número de gaussianas es fijo, tanto para el modelo existente como para el modelo incrementado resultado de las técnicas propuestas. Además, dicho número es escogido de manera heurística para cada prueba, y se supone adecuado para modelar correctamente todas las trayectorias.

6.2.1. Técnica Generativa

A partir de la técnica generativa para GMM propuesta en [27], se reescribieron las ecuaciones para que trataran ahora con modelos de mezclas de gaussianas parametrizado en la tarea. Con el modelo existente, se generan las trayectorias correspondientes con los parámetros de la tarea que se utilizó para estimar el modelo. Con estas trayectorias y las nuevas que se desean agregar, se forma un conjunto, con el cual se estima un nuevo modelo TPGMM. La técnica consta de los siguientes pasos:

1. A partir del modelo existente, se genera igual número de trayectorias. Se supone que los parámetros de la tarea de las trayectorias anteriores, son conocidos.

2. A las trayectorias generadas se le agregan las nuevas trayectorias, generando un nuevo conjunto de datos.

3. Con el conjunto de datos obtenido en el paso anterior, se calculan los parámetros iniciales del modelo.

4. Se estima un nuevo modelo, usando el algoritmo EM modificado, el mismo de TPGMM.

Dado el modelo PGMM parametrizado en la tarea y los parámetros de la tarea $(\boldsymbol{A}, \boldsymbol{b})$, se utiliza regresión de mezclas gaussianas (GMR) para reproducir las M trayectorias, a las cuales se les aplica un filtro de promedio móvil, obteniendo las trayectorias generadas $\boldsymbol{\xi_g}$. Estas se agrupan con la o las trayectorias que se desean agregar $\boldsymbol{\xi_n}$:

$$\begin{aligned} \boldsymbol{\xi}_c &= [\boldsymbol{\xi}_g, \boldsymbol{\xi}_n], \\ \boldsymbol{A}_c &= [\boldsymbol{A}, \boldsymbol{A}_n], \\ \boldsymbol{b}_c &= [\boldsymbol{b}, \boldsymbol{b}_{new}]. \end{aligned} \tag{6-1}$$

Con la información anterior se estima un nuevo modelo inicial, usando la técnica de división de los datos en el tiempo. Con el modelo obtenido de la inicialización, se estiman los parámetros del modelo definitivos, usando el algoritmo EM modificado presentado en la Sección 3.3.

6.2.2. Técnica de Adición de Modelos

Esta permite adicionar un modelo existente, al modelo estimado de las trayectorias nuevas que se desean agregar. Se basa en la adición de modelos de mezcla de gaussianas (GMM), propuesto en el 2004 por Hall y Hikcs [54]. La modificación consistió en convertir las ecuaciones de modelos de mezcla de gaussianas, a ecuaciones de modelos de mezclas de gaussianas parametrizado en la tarea, la técnica consta de los siguientes pasos:

1. Se calcula el modelo de la o las nuevas trayectorias.

2. Se concatena el modelo obtenido en el paso anterior con el modelo existente.

3. Se simplifica el modelo resultado de la concatenación, usando un proceso de optimización.

Concatenación: Dados dos modelos TPGMM: $\{\alpha_i, \boldsymbol{Z}^{\nu}_{i,j}, \boldsymbol{Z}^{C}_{i,j}\}$, $\{\grave{\alpha}_i, \grave{\boldsymbol{Z}}^{\nu}_{i,j}, \grave{\boldsymbol{Z}}^{C}_{i,j}\}$ con probabilidad de distribución:

$$p(\boldsymbol{\xi}) = \sum_{i=1}^{K_1} \alpha_i \mathcal{N}(\boldsymbol{\xi} | \boldsymbol{Z}^{\nu}_{i,j}, \boldsymbol{Z}^{C}_{i,j}), \tag{6-2}$$

$$q(\boldsymbol{\xi}) = \sum_{i=1}^{K_2} \grave{\alpha}_i \mathcal{N}(\boldsymbol{\xi} | \grave{\boldsymbol{Z}}^{\nu}_{i,j}, \grave{\boldsymbol{Z}}^{C}_{i,j}). \tag{6-3}$$

Para $j = 1, ..., N_P$ marcos de referencia, la suma de las probabilidades es:

$$\begin{aligned} r(\boldsymbol{\xi}) &= f_1 p(\boldsymbol{\xi}) + f_2 q(\boldsymbol{\xi}), \\ r(\boldsymbol{\xi}) &= f_1 \sum_{i=1}^{K_1} \alpha_i \mathcal{N}(\boldsymbol{\xi} | \boldsymbol{Z}_{i,j}^{\nu}, \boldsymbol{Z}_{i,j}^{C}) + f_2 \sum_{i=1}^{K_2} \grave{\alpha}_i \mathcal{N}(\boldsymbol{\xi} | \grave{\boldsymbol{Z}}_{i,j}^{\nu}, \grave{\boldsymbol{Z}}_{i,j}^{C}), \\ r(\boldsymbol{\xi}) &= \sum_{i=1}^{K_1+K_2} \beta_i \mathcal{N}(\boldsymbol{\xi} | \boldsymbol{Z}_{i,j}^{\vartheta}, \boldsymbol{Z}_{i,j}^{\Xi}). \end{aligned} \tag{6-4}$$

$$\begin{aligned} \boldsymbol{\beta} &= [f_1 \boldsymbol{\alpha}, f_2 \grave{\boldsymbol{\alpha}}], \\ \boldsymbol{Z}^{\vartheta} &= [\boldsymbol{Z}^{\nu}, \grave{\boldsymbol{Z}}^{\nu}], \\ \boldsymbol{Z}^{\Xi} &= [\boldsymbol{Z}^{C}, \grave{\boldsymbol{Z}}^{C}] \end{aligned} \tag{6-5}$$

La Ecuación (6-4) equivale al modelo concatenado $\{\beta_i, \boldsymbol{Z}_{i,j}^{\vartheta}, \boldsymbol{Z}_{i,j}^{\Xi}\}$, el cual tiene $(K_1 + K_2)$ componentes, donde $f_2 = 1 - f_1$.

Simplificación: Dado los elementos del modelo concatenado, este se puede simplificar a un modelo con K componentes, con $K < K_1 + K_2$. La simplificación se realiza a partir de una matriz de mezcla, entre el modelo concatenado y el simplificado $\boldsymbol{w}$:

$$\pi_l = \sum_{i=1}^{K_1+K_2} w_{i,l} \beta_i, \tag{6-6}$$

$$\boldsymbol{Z}_{l,j}^{\mu} = \frac{1}{\pi_l} \sum_{i=1}^{K_1+K_2} w_{i,l} \beta_i \boldsymbol{Z}_{i,j}^{\vartheta}, \tag{6-7}$$

$$\boldsymbol{Z}_{l,j}^{\Sigma} = \frac{1}{\pi_l} \left(\sum_{i=1}^{K_1+K_2} w_{i,l} \beta_i \left(\boldsymbol{Z}_{i,j}^{\Xi} + \boldsymbol{Z}_{i,j}^{\vartheta} (\boldsymbol{Z}_{i,j}^{\vartheta})^T \right) \right) - \boldsymbol{Z}_{l,j}^{\mu} (\boldsymbol{Z}_{l,j}^{\mu})^T, \tag{6-8}$$

Con las restricciones:

$$\sum_{i=1}^{K_1+K_2} w_{i,j} = 1, \tag{6-9}$$

$$\sum_{i=1}^{K_1+K_2} w_{i,j} \alpha_i < 1. \tag{6-10}$$

Contrario al algoritmo de optimización descrito en [54], para hallar la matriz de mezcla $\boldsymbol{w}$, el cual presenta una convergencia lenta, se utilizó una función de optimización basada en el error entre las trayectorias generadas por el modelo concatenado y el modelo simplificado. A partir de un $\boldsymbol{w}$

inicial y dicha función que se desea minimizar, se hallan los pesos que permiten obtener el modelo simplificado. El valor inicial de $\boldsymbol{w}$ se seleccionó de manera heurística, como:

$$\boldsymbol{w} = 0{,}45 \begin{bmatrix} I_{K_1 \times K} \\ I_{K_2 \times K} \end{bmatrix}, \tag{6-11}$$

donde I es la matriz identidad.

La función a minimizar que se utilizó, está dada por:

$$f = \frac{1}{T_m} \sum_{m=1}^{M} \sum_{k=1}^{T_m} \|\boldsymbol{y}_m(k) - \boldsymbol{y}_m^s(k)\|^2, \tag{6-12}$$

donde $\boldsymbol{y}_m$ es el conjunto formado por las trayectorias reproducidas usando el modelo anterior y las trayectorias demostradas que se desean agregar, y $\boldsymbol{y}_m^s$ son las trayectorias equivalentes, reproducidas con el modelo simplificado a partir de los parámetros de la tarea, las cuales se recalculan en cada iteración del proceso de minimización. Para la implementación del algoritmo se utilizó la función para Matlab *fminsearchcon*, sujeta a las restricciones descritas en las ecuaciones (6-9) y (6-10).

6.2.3. Técnica de Actualización Directa

A partir de la técnica de actualización directa para GMM propuesta en [27], se modificaron las ecuaciones de modelos GMM, a ecuaciones de modelos GMM parametrizados en la tarea. Con el modelo existente, se generan las trayectorias correspondientes con los parámetros de la tarea que se utilizó para estimar el modelo. En esta técnica, en las ecuaciones del algoritmo de esperanza maximización (EM), se separan los componentes correspondientes a los parámetros del modelo existente y los componentes relacionados con las nuevas trayectorias. La técnica asume que el conjunto de probabilidades posteriores es similar tanto para las trayectorias pasadas como para las nuevas. Definiendo los parámetros del modelo TPGMM existente $\{\boldsymbol{\pi}, \boldsymbol{Z}^{\mu}, \boldsymbol{Z}^{\Sigma}\}$ con K gaussianas, y las nuevas trayectorias $\tilde{\boldsymbol{\xi}}$ a incrementar, compuestas por T_m muestras y $\tilde{M}$ trayectorias de demostración, se obtienen $\tilde{N} = \tilde{M}T_m$ puntos de datos de dimensión $D + 1$. A partir de los parámetros del modelo TPGMM existente y las nuevas trayectorias, se actualiza el modelo existente con los siguientes pasos:

1. Cálculo del vector de esperanza inicial con los coeficientes de mezcla inicial y la cantidad de puntos de datos existentes N:

   $$E_i = N\pi_i \tag{6-13}$$

2. Con las trayectorias que se desean agregar, se estima un modelo TPGMM inicial de las nuevas trayectorias: $\{\tilde{\boldsymbol{\pi}}, \tilde{\boldsymbol{Z}}^{\mu}, \tilde{\boldsymbol{Z}}^{\Sigma}\}$, usando la técnica de división de los datos en el tiempo.

3. Se aplica el algoritmo EM modificado, o método directo, un número dado adicional de veces:

   Paso E:

$$\tilde{h}_{n,i} = \frac{\tilde{\pi}_i \mathcal{N}(\tilde{\boldsymbol{\xi}}_n | \tilde{\boldsymbol{\mu}}_{n,i}, \tilde{\boldsymbol{\Sigma}}_{n,i})}{\sum_k^{N_K} \tilde{\pi}_k \mathcal{N}(\tilde{\boldsymbol{\xi}}_n | \tilde{\boldsymbol{\mu}}_{n,k}, \tilde{\boldsymbol{\Sigma}}_{n,k})}, \tag{6-14}$$

   donde $\tilde{\boldsymbol{\mu}}_{n,i}$ y $\tilde{\boldsymbol{\Sigma}}_{n,i}$ es calculada como en la Ecuación (3-40).

   Paso M:

$$\begin{aligned}
\tilde{E}_i &= \sum_{n=1}^{\tilde{N}} \tilde{h}_{n,i}, \\
\tilde{\pi}_i &= \frac{E_i + \tilde{E}_i}{N + \tilde{N}}, \\
\tilde{\boldsymbol{Z}}_{i,j}^{\mu} &= \frac{E_i \boldsymbol{Z}_{i,j}^{\mu} + \sum_{n=1}^{\tilde{N}} \tilde{h}_{n,i} \tilde{\boldsymbol{A}}_{n,j}^{-1} [\tilde{\boldsymbol{\xi}}_n - \tilde{\boldsymbol{b}}_{n,j}]}{E_i + \tilde{E}_i}, \\
\tilde{\boldsymbol{Z}}_{i,j}^{\Sigma} &= \frac{\boldsymbol{s}_1 + \boldsymbol{s}_2}{E_i + \tilde{E}_i},
\end{aligned} \tag{6-15}$$

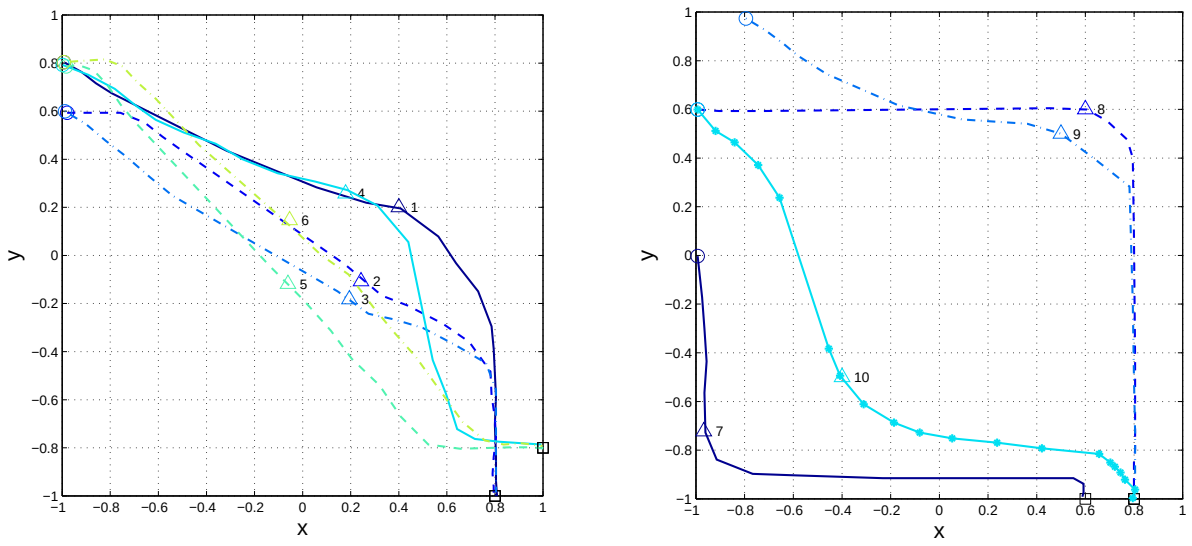
   con:

$$\begin{aligned}
\boldsymbol{s}_1 &= E_i [\boldsymbol{Z}_{i,j}^{\Sigma} + (\boldsymbol{Z}_{i,j}^{\mu} - \tilde{\boldsymbol{Z}}_{i,j}^{\mu})(\boldsymbol{Z}_{i,j}^{\mu} - \tilde{\boldsymbol{Z}}_{i,j}^{\mu})^T], \\
\boldsymbol{s}_2 &= \sum_{n=1}^{\tilde{N}} \tilde{h}_{n,i} \tilde{\boldsymbol{A}}_{n,j}^{-1} [\tilde{\boldsymbol{\xi}}_n - \hat{\boldsymbol{\mu}}_{n,i,j}][\tilde{\boldsymbol{\xi}}_n - \hat{\boldsymbol{\mu}}_{n,i,j}]^T \tilde{\boldsymbol{A}}_{n,j}^{-T}, \\
\hat{\boldsymbol{\mu}}_{n,i,j} &= \tilde{\boldsymbol{A}}_{n,j} \tilde{\boldsymbol{Z}}_{i,j}^{\mu} + \tilde{\boldsymbol{b}}_{n,j}.
\end{aligned}$$

6.3. Pruebas y Resultados

Las técnicas propuestas se validaron mediante dos pruebas, la primera consiste en la simulación de la tarea de barrer y la segunda es la tarea de colocar la manga de una camisa, en el brazo de un maniquí, usando un robot WAM. La reproducción del modelo TPGMM para la técnica existente y las propuestas son realizadas con GMR [115] y [4], y las trayectorias resultantes son filtradas con un filtro de promedio móvil de dos muestras, con esto se busca suavizar las trayectorias y que en consecuencia el robot tenga un movimiento más suave. Para la comparación de las trayectorias que se dan en cada tarea se utilizó el error RMS, entre una trayectoria demostrada $\boldsymbol{\xi}$ y la trayectoria reproducida $\boldsymbol{y}$:

$$e_{RMS} = \frac{1}{T} \sum_{k=1}^{T} \|\boldsymbol{y}(k) - \boldsymbol{\xi}(k)\|, \tag{6-16}$$



(a) Iniciales. (b) Incrementadas.

Figura 6-1: Trayectorias de la prueba de simulación. a) Utilizadas para el cálculo del modelo existente (trayectorias 1 a 6). b) Utilizadas para incrementar (trayectorias 7 a 10). El círculo indica el punto de inicio de la trayectoria, el triángulo la posición del objeto a barrer y el cuadrado la posición del recogedor.

donde T es el total de muestras de la demostración. Con esta métrica se compara:

- La reproducción usando el modelo existente estimado con las trayectorias iniciales, contra los obtenidos con las técnicas propuestas incrementadas en la trayectoria dada.

- La reproducción usando un modelo TPGMM calculado con el conjunto de las trayectorias iniciales y la o las trayectorias a incrementar, contra el obtenido con una de las técnicas propuestas incrementado con una trayectoria dada.

Las pruebas pretenden visualizar y analizar las ventajas y desventajas de las técnicas propuestas. En la técnica de adición de modelos, tanto para la simulación como para la tarea de colocar la manga, el valor de f_1 se fijó como $0{,}5$, con esto se busca una combinación equitativa de los parámetros de los dos modelos.

Por simplicidad y fácil entendimiento de las figuras, y considerando que el desempeño de la técnica directa no es alto, los resultados de esta técnica solo se muestran en la prueba de adición de múltiples trayectorias y en la Figura **6-17** de la Sección de discusión.

6.3.1. Simulación de la Tarea de Barrer Objetos

Se simuló la tarea de barrer un objeto, la cual consiste en que a partir de un punto inicial, el efector final se desplaza hasta el objeto a barrer y éste se lleva a un punto recogedor [4]. Las trayectorias se obtuvieron manualmente usando el ratón del computador, estas se muestran en la Figura **6-1**. Los puntos que conforman la trayectoria se re-muestrearon usando la función "spline" de Matlab. Para la estimación del modelo existente se usaron las trayectorias de la Figura **6-1**a, y para las pruebas se agregó las trayectorias de la Figura **6-1**b. Los parámetros de la tarea que se tomaron fueron: El punto de inicio, la ubicación del objeto y la ubicación del recogedor. A partir de estas posiciones, se calculó la matriz $\boldsymbol{A}$. El vector $\boldsymbol{b}$, se obtuvo a partir de dos puntos ($\boldsymbol{p}_1$ y $\boldsymbol{p}_2$), en el espacio cartesiano. El punto $\boldsymbol{p}_1$ está relacionado con alguno de los parámetros de tarea: inicio, ubicación objeto, final; el punto $\boldsymbol{p}_2$, se tomó como la tercera muestra a partir del punto $\boldsymbol{p}_1$.

La matriz de transformación $\boldsymbol{A}$, se forma a partir de tres vectores $\boldsymbol{v}_1$, $\boldsymbol{v}_2$, y $\boldsymbol{v}_3$.

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & v_{1x} & v_{2x} & v_{3x} \\ 0 & v_{1y} & v_{2y} & v_{3y} \\ 0 & v_{1z} & v_{2z} & v_{3z} \end{bmatrix}. \tag{6-17}$$

El vector $\boldsymbol{v}_2$, es el vector normal a la diferencia $\boldsymbol{p}_2 - \boldsymbol{p}_1$, y los vectores $\boldsymbol{v}_1$ y $\boldsymbol{v}_3$ son perpendiculares a $\boldsymbol{v}_2$, con lo que $\boldsymbol{A}$ es una matriz de transformación que contiene información de la orientación del punto $\boldsymbol{p}_1$ con respecto al punto $\boldsymbol{p}_2$. En el caso del parámetro de ubicación del objeto, esta matriz es la identidad. El vector de desplazamiento $\boldsymbol{b}$ se forma a partir del punto $\boldsymbol{p}_1$, como:

$$\boldsymbol{b} = [0, p_{1x}, p_{1y}, p_{1z}]^T. \tag{6-18}$$

El número de estados del modelo se asumió en cuatro, y es igual tanto para el cálculo del modelo existente, como para el de las técnicas generativa y adición de modelos, este valor se ajustó de manera heurística.

En la Figura **6-2** se muestra un ejemplo de las reproducciones usando los parámetros de la tarea correspondientes a la trayectoria siete (línea en color amarillo). En la Figura **6-2** también se presenta la repuesta del modelo (línea a tramos en color negro), calculado con las trayectorias existentes y la respuesta con el modelo calculado con la técnica generativa (color verde), y en rojo la respuesta con la técnica de adición de modelos. Al usar las técnicas propuestas, se consigue una mejoría con respecto a la reproducción obtenida con el modelo calculado con solo las trayectorias existentes.

En la Figura **6-3** se muestra una comparación del error RMS usando el modelo calculado con las trayectorias iniciales, con el obtenido al agregar cada una de las trayectorias (7 a 10). En todas las

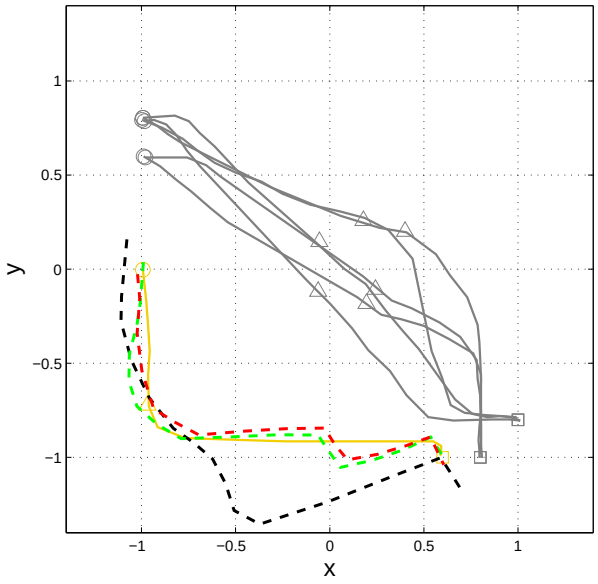


Figura 6-2: Ejemplo de la reproducción de la trayectoria siete (color amarillo). Línea Negro a tramos: Modelo existente. Verde a tramos: Generativa. Rojo a tramos: Adición de modelos. Gris continua: Demostraciones iniciales.

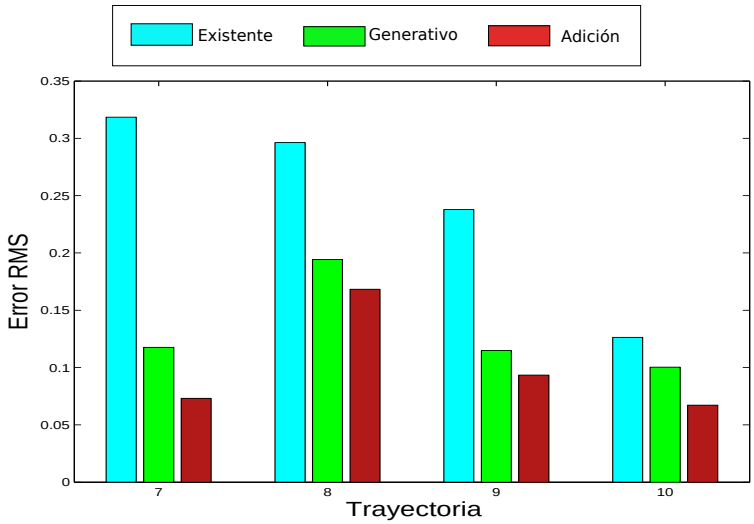


Figura 6-3: Error RMS para la tarea simulada y comparando el modelo existente contra los modelos incrementados. Azul: TPGMM existente. Verde: Generativa. Rojo: Adición de modelos.

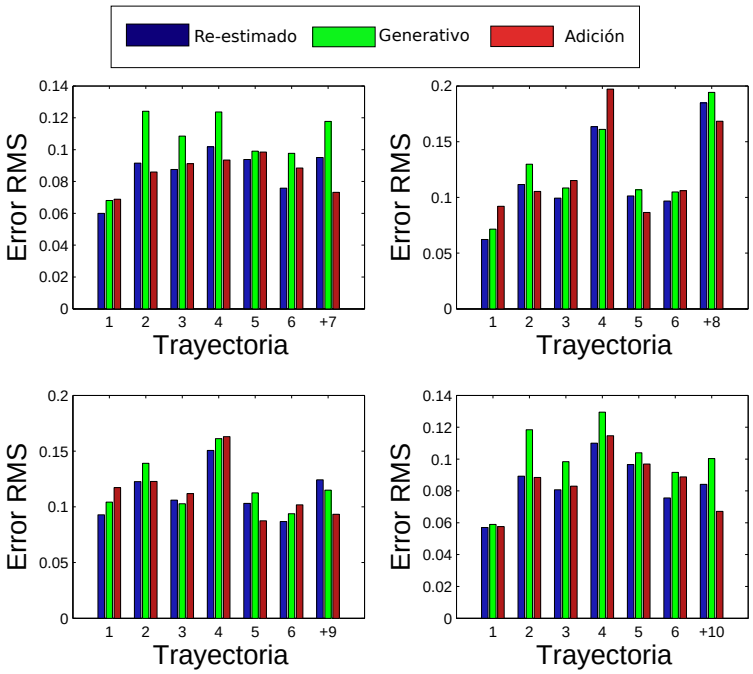


Figura 6-4: Error RMS para la tarea simulada y comparando el modelo TPGMM incrementado contra los modelos incrementados. Azul: Modelo TPGMM incrementado. Verde: Técnica generativa. Rojo: Adición de modelos.

trayectorias agregadas, se puede notar una disminución del error al usar las técnicas propuestas. La técnica de adición de modelos es la que presenta el menor error en la trayectoria incrementada, lo que se debe a dos factores: *i)* La reconstrucción en el caso de la técnica generativa conlleva a errores y *ii)* La optimización implicada en la técnica de adición hace que el error RMS disminuya. La varianza para cada error no es presentada en la figura, debido a que las técnicas propuestas no dependen de valores aleatorios en la inicialización o estimación, con lo que la estimación del error siempre arroja el mismo resultado, cada vez que se ejecuta la técnica.

En la Figura **6-4** se muestra una comparación del error RMS de la técnica TPGMM incrementada con una trayectoria dada, con los obtenidos usando las técnicas propuestas. En esta se puede observar que, excepto unos pocos casos, el error es similar y que la técnica de adición de modelos presenta el menor error de las dos técnicas incrementales para las trayectorias iniciales (1 a 6). Es de anotar que el error obtenido con la técnica TPGMM, es calculado con la información de las trayectorias demostradas tanto existentes como agregadas, lo cual hace que su error sea menor. En este caso, solo se agregó una trayectoria a la vez, el siguiente ejemplo, presenta la adición de múltiples trayectorias.

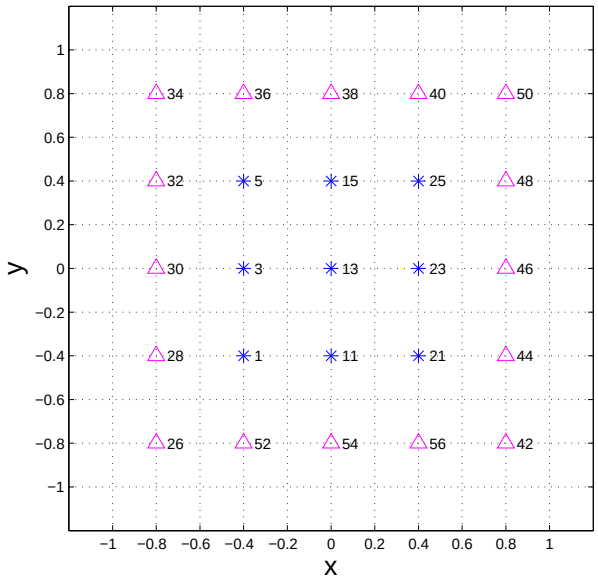
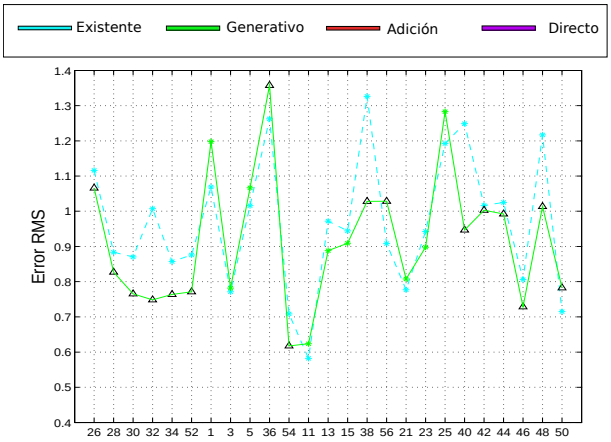


Figura 6-5: Diversas posiciones del objeto a barrer. Las posiciones marcadas con asterisco, se refieren a trayectorias usadas para estimar un modelo TPGMM inicial y las marcadas con triángulos corresponden a las trayectorias que se agregan.

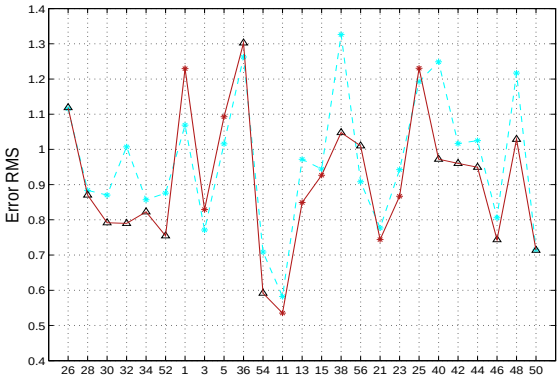
Prueba Agregando Múltiples Trayectorias.
En la Figura **6-5** se muestra la cuadrícula al variar la posición del objeto a barrer, dejando fijas la posición de la cual inician y finalizan las trayectorias. Por cada posición de la cuadrícula se realizó manualmente una trayectoria. En esta prueba se presentan resultados de las técnicas generativa, adición de modelos, y actualización directa.

El modelo existente se estimó con la cuadrícula central (asteriscos numerados con valores menores o iguales a 25), y para la estimación de los nuevos modelos, se usó el cuadrado externo (triángulos numerados con valores mayores a 25). En la Figura **6-6**a se muestra el error RMS entre cada trayectoria demostrada y la reproducida usando el modelo TPGMM existente (color azul claro), en verde el error RMS usando la reproducción con la técnica generativa, de manera similar en la Figura **6-6**b en rojo la reproducción con la técnica de adición de modelos, y en la Figura **6-6**c en magenta con la técnica de actualización directa. Las trayectorias agregadas se muestran con un triángulo negro, en casi todos los casos de trayectorias agregadas, el error es menor. Es de anotar que el error no se reduce significativamente como en el caso anterior, debido a que el error del modelo existente es mucho menor, ya que este se estima con 9 trayectorias (cuadrícula central marcada con asteriscos) y las trayectorias externas son similares a las centrales.

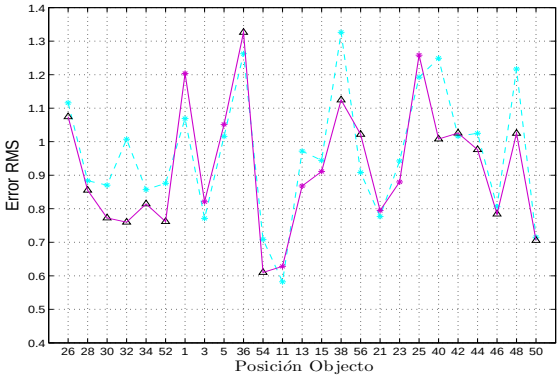
En la Figura **6-7** se muestran las reproducciones usando los modelos resultado de las tres técnicas, con los parámetros correspondientes a la trayectoria inicial número uno. En línea negra a tramos,



(a) Existente y Generativa.



(b) Existente y Adición de modelos.



(c) Existente y Actualización Directa.

Figura 6-6: Error RMS para las diversas posiciones del objeto a barrer y las técnicas propuestas. Las posiciones marcadas con asterisco se refieren a trayectorias usadas para estimar un modelo TPGMM inicial (color azul), y las marcadas con triángulos corresponden a las trayectorias que se agregan.

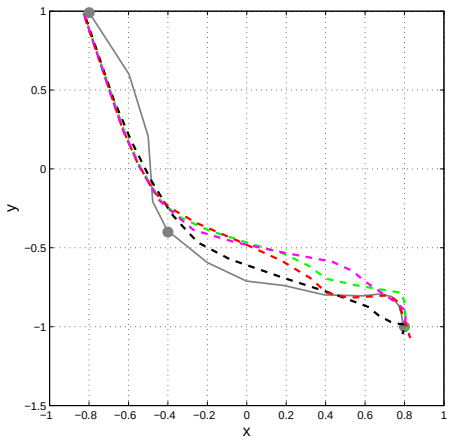


Figura 6-7: Ejemplo de la trayectoria de respuesta usando los parámetros de la tarea para la trayectoria 1. Línea Negro a tramos: Modelo existente. Verde a tramos: Generativa. Rojo a tramos: Adición de modelos. Magenta a tramos: Actualización directa.

se muestra el resultado con el modelo existente, en línea verde a tramos el resultado con la técnica generativa, en rojo con la técnica de adición de modelos y en magenta la técnica de actualización directa. Este último es el que más se aleja de la trayectoria original (línea continua en color gris).

6.3.2. Tarea Real

Se implementó la tarea de colocar la manga de una camisa en el brazo de un maniquí, esta implica que existen diversas trayectorias, al variar la posición y curvatura del brazo del maniquí, tanto cuando el brazo está adelante como cuando está atrás. En la Figura **6-8** se muestran dos ejemplos de posiciones del brazo y, en la Figura **6-9**, las trayectorias obtenidas a partir de demostraciones cinestáticas de estas. Se estimó un modelo TPGMM con las trayectorias de brazo adelante, obteniendo lo que denominamos modelo existente, y se estimó con las técnicas generativa y adición de modelos, incrementando con cada una de las trayectorias de brazo atrás, obteniendo los modelos resultado de las técnicas.

Se seleccionaron los siguientes parámetros de la tarea: Punto sobre la mano, Punto sobre el codo, Punto sobre el hombro. En la Figura **6-10**, se muestra la forma como se tomaron las lecturas de los parámetros de la tarea, lo cual consiste en colocar el efector final de robot en el punto medido. En todas las medidas el efector final se orientó perpendicular al suelo, buscando que luego fuera sencillo tomar nuevos parámetros. Para el cálculo de la matriz $\boldsymbol{A}$ (dirección) y el vector de desplazamiento $\boldsymbol{b}$ (posición), se utilizó la información de posición y orientación del efector final, a partir de dos puntos ($\boldsymbol{p}_1$ y $\boldsymbol{p}_2$), en el espacio cartesiano. El punto $\boldsymbol{p}_1$ está relacionado con alguno de los parámetros de la tarea, en la Tabla **6-1** se muestran los puntos $\boldsymbol{p}_1$ y $\boldsymbol{p}_2$ para cada caso. La

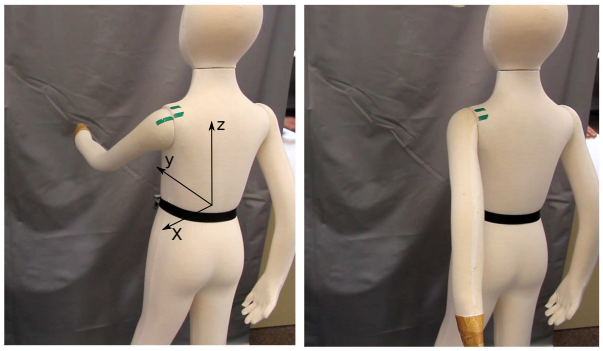


Figura 6-8: Dos ejemplos de posición del brazo. Izquierda: Brazo adelante. Derecha: Brazo atrás.

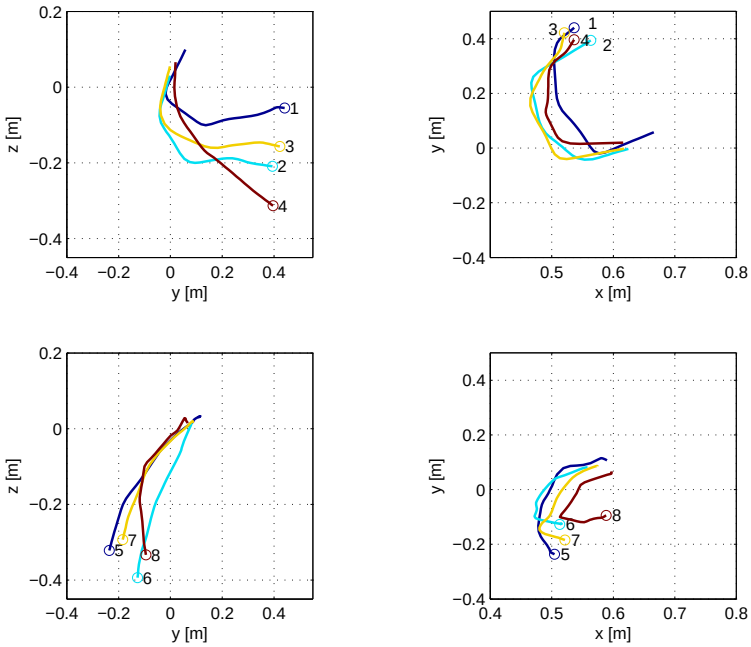
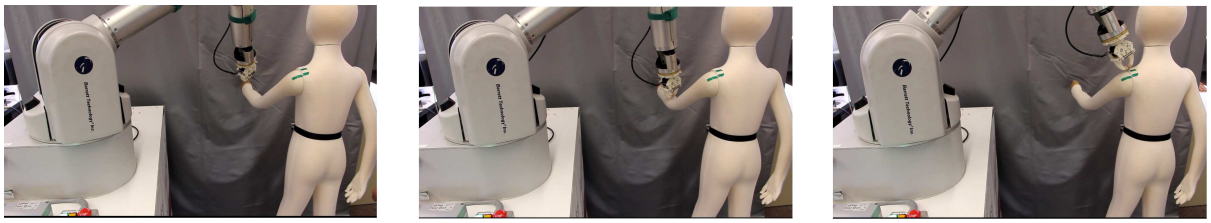


Figura 6-9: Trayectorias utilizadas. Superior: Brazo adelante. Inferior: Brazo atrás.



(a) Mano. (b) Codo. (c) Hombro.

Figura 6-10: Obtención de los parámetros de la tarea ubicando el efector final del robot sobre un punto.

Tabla **6-1**: Valores de $\boldsymbol{p}_1$ y $\boldsymbol{p}_2$ según el parámetro de la tarea.

	$\boldsymbol{p}_1$	$\boldsymbol{p}_2$
Mano	posición mano	posición codo
Codo	Posición codo	-
Hombro	posición hombro	posición codo

variable $\boldsymbol{b}$ de los parámetros de la tarea está descrito por 8 valores:

$$\boldsymbol{b} = \begin{bmatrix} t \\ x \\ y \\ z \\ qx \\ qy \\ qz \\ qw \end{bmatrix}, \tag{6-19}$$

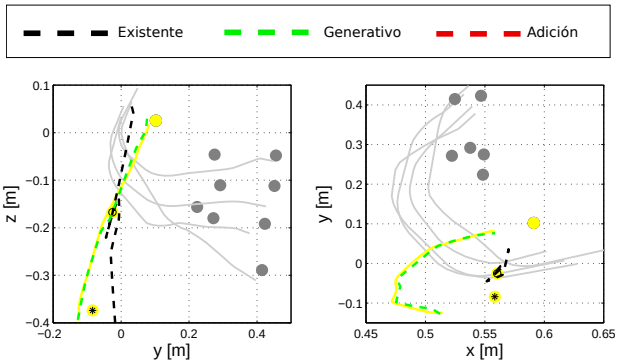
donde t es el tiempo, x, y, z es la posición cartesiana y qx, qy, qz, qw es la orientación en cuaternios.

Para el cálculo de $\boldsymbol{A}$, solo se usan los valores de posición (x, y, z).

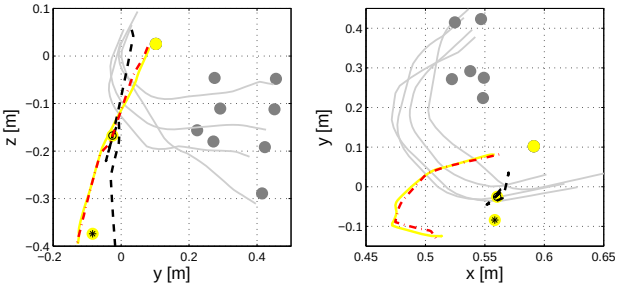
$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & v_{1x} & v_{2x} & v_{3x} & 0 & 0 & 0 & 0 \\ 0 & v_{1y} & v_{2y} & v_{3y} & 0 & 0 & 0 & 0 \\ 0 & v_{1z} & v_{2z} & v_{3z} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{6-20}$$

De nuevo, el vector $\boldsymbol{v}_2$, es el vector normal a la diferencia $\boldsymbol{p}_2 - \boldsymbol{p}_1$ y los vectores $\boldsymbol{v}_1$ y $\boldsymbol{v}_3$, son perpendiculares a $\boldsymbol{v}_2$. Con lo que $\boldsymbol{A}$ es una matriz de transformación, que contiene información de la orientación del punto $\boldsymbol{p}_1$ con respecto al punto $\boldsymbol{p}_2$.

Para la reproducción de las trayectorias, tanto de la técnica TPGMM como de las técnicas propuestas, se utilizaron cuatro estados. En la Figura **6-11**, se muestra un ejemplo de las reproducciones usando los parámetros correspondientes a la trayectoria seis (línea en color verde). En



(a) Técnica generativa.



(b) Técnica de adición de modelos.

Figura 6-11: Trayectorias reproducidas usando el modelo estimado con las trayectorias de brazo adelante y los modelos obtenidos con las técnicas propuestas. Trayectoria demostrada número 6. (color Amarillo), reproducción usando el modelo estimado con las trayectorias de brazo adelante (línea a tramos color negro). Línea Verde a tramos con la técnica generativa, en rojo a tramos con la técnica de adición de modelos. En gris, las trayectorias demostradas iniciales.

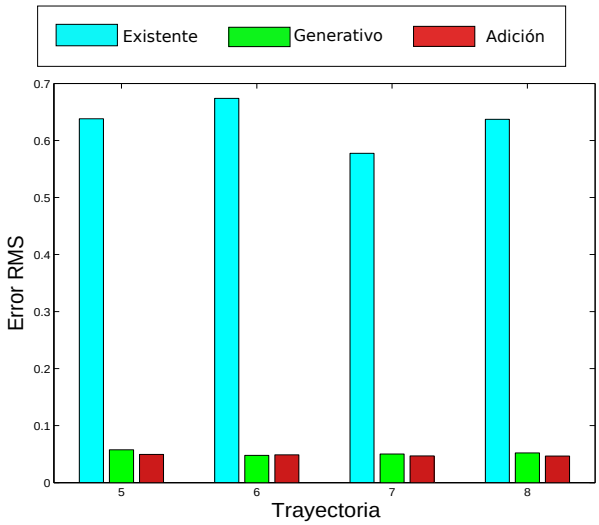


Figura 6-12: Error RMS para la tarea real y comparando el modelo existente contra los modelos incrementados con cada trayectoria (5 a 8). Azul oscuro: Estimado con las trayectorias de mano adelante. Verde: Generativa. Rojo: Adición de modelos.

la Figura **6-11**a, se muestra la repuesta del modelo estimado con las trayectorias de mano adelante (línea a tramos en color negro) y con la técnica generativa (color verde). En la Figura **6-11**b, en rojo, se presenta la técnica de adición de modelos. Similar que en la prueba de simulación presentada anteriormente, al usar las técnicas propuestas, se consigue una mejora en comparación con la reproducción obtenida cuando se usa el modelo calculado con solo las trayectorias existentes.

En la Figura **6-12** se muestra una comparación del error RMS para la tarea real, usando el modelo estimado con las trayectorias iniciales, contra el obtenido al agregar cada una de las trayectorias (5 a 8). En todas las trayectorias agregadas se puede notar una disminución del error al usar las técnicas propuestas, en este caso las técnicas propuestas dan un valor similar de error, al incrementar cada una de las trayectorias de mano atrás.

En la Figura **6-13** se muestra una comparación del error RMS para la tarea real, usando la técnica TPGMM, estimado con el conjunto de las trayectorias iniciales y una trayectoria dada, contra el obtenido usando las técnicas propuestas. Los errores RMS de las técnicas propuestas, presentan valores similares o ligeramente superiores al obtenido con el modelo TPGMM incrementado, el cual es estimado con las trayectorias demostradas iniciales y agregadas. La Figura **6-14** muestra dos secuencias de imágenes tomadas del robot colocando la manga. Superior) Con el brazo del maniquí hacia atrás. Inferior) Con el brazo del maniquí hacia adelante.

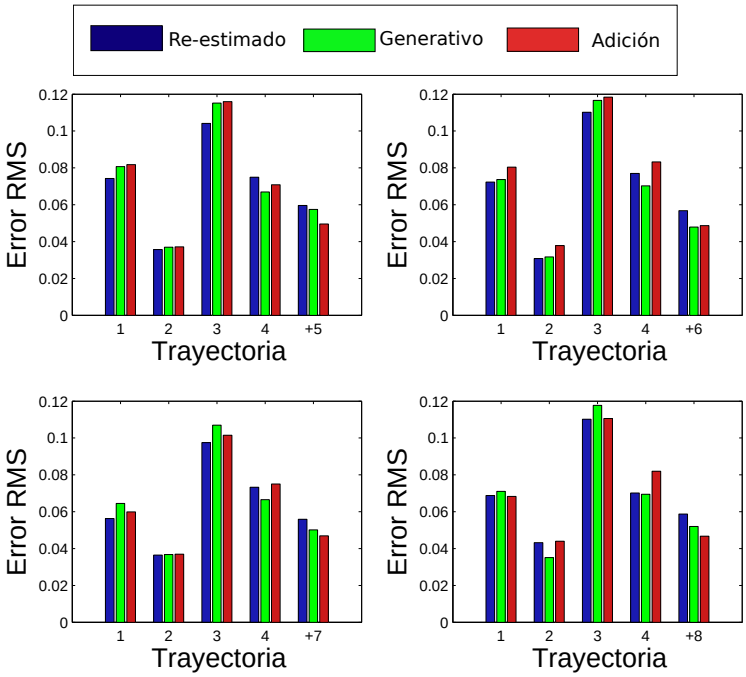


Figura 6-13: Error RMS para la tarea real y comparando el modelo TPGMM incrementado contra los modelos incrementados. Azul oscuro: Estimado con las trayectorias de mano adelante. Verde: Generativa. Rojo: Adición de modelos.

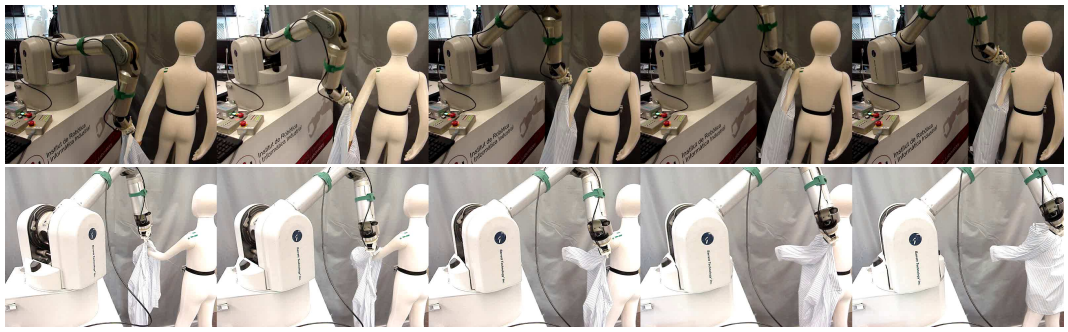


Figura 6-14: Secuencia de imágenes del robot colocando la manga. Superior: con el brazo del maniquí hacia atrás. Inferior: con el brazo del maniquí hacia adelante.

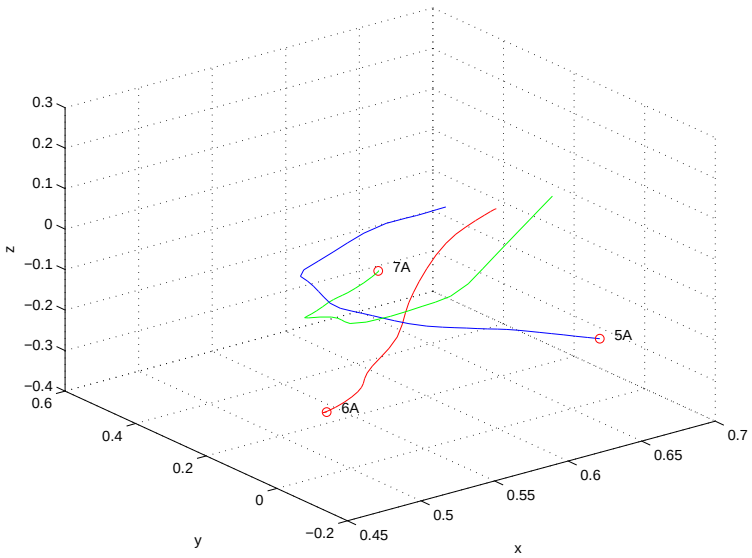


Figura 6-15: Trayectorias con parámetros alejados.

Trayectorias con Parámetros de la Tarea más Alejados.
Se usaron tres trayectorias de la tarea de colocar la manga, cuyos parámetros están alejados de los de mano adelante y de los de mano atrás, estas trayectorias se muestran en la Figura **6-15**. En este caso, el modelo TPGMM existente es estimado con las trayectorias de mano adelante y las técnicas generativa y adición de modelos, se estiman con las trayectorias de mano adelante y una de las mostradas de parámetros alejados. En la Figura **6-16** se muestra el error RMS obtenido con el modelo TPGMM estimado con el conjunto formado con las trayectorias de mano adelante y una de las tres trayectorias mostradas en la Figura **6-15**, contra el obtenido con las técnicas propuestas incrementado con cada una de las trayectorias mencionadas. En la Figura **6-16** se observa que la técnica generativa es la que presenta el error más bajo entre las técnicas propuestas, para las trayectorias iniciales.

6.3.3. Discusión

En relación al tiempo de estimación de las técnicas, usando un computador Intel de 2 núcleos a 1.3GHz corriendo Matlab, la técnica de actualización directa toma 10 segundos, la técnica generativa toma en promedio 20 segundos, y la técnica de adición de modelos tarda en promedio 80 segundos para las trayectorias simuladas y 120 segundos para las trayectorias de colocar la manga de la camisa. Aunque los tiempos de está última son grandes, son mucho menores de lo que tomaría en entrenar con el robot todas las trayectorias, en caso de no contar con ellas.

En la Figura **6-17**, se muestran los diagramas de barras para todas las pruebas, los resultados se

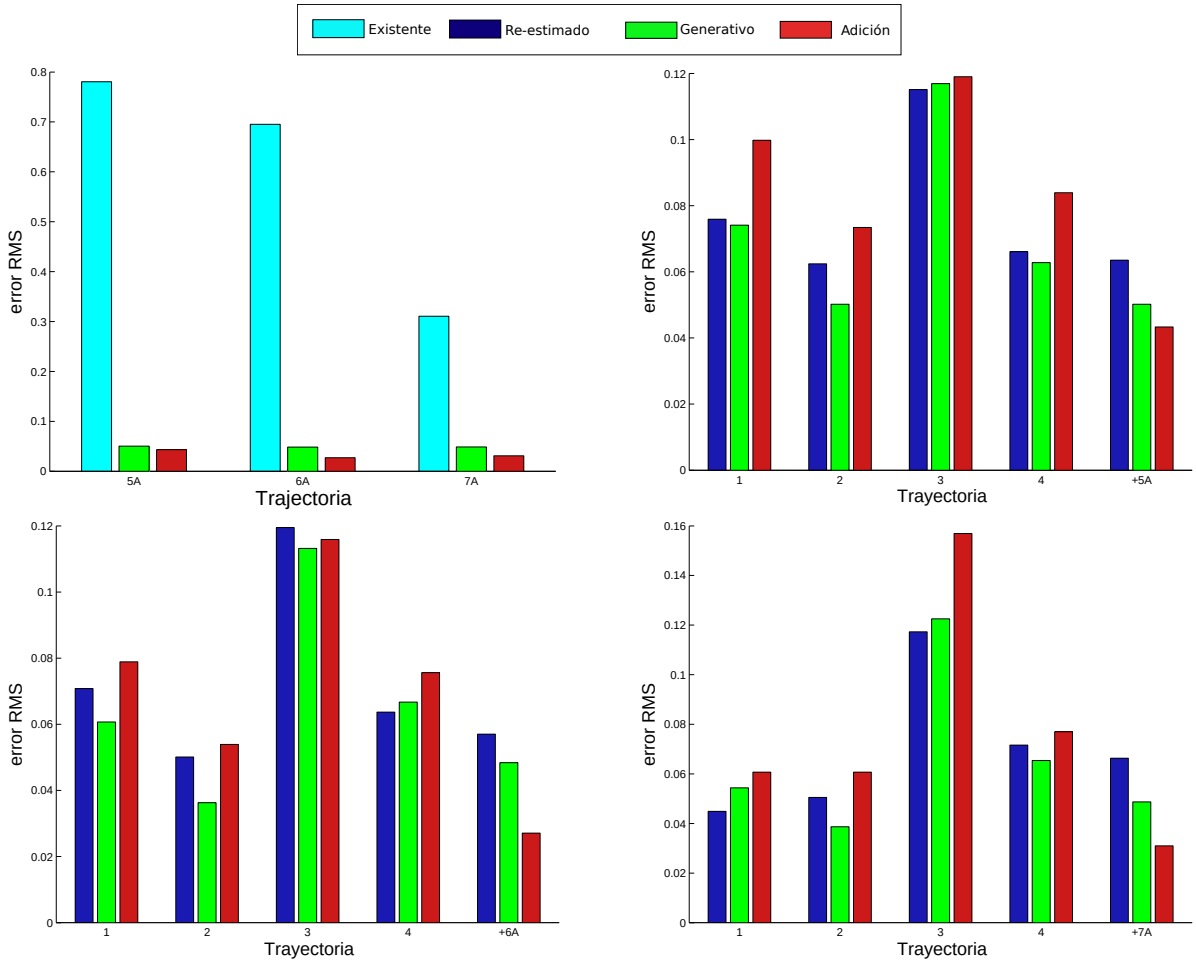
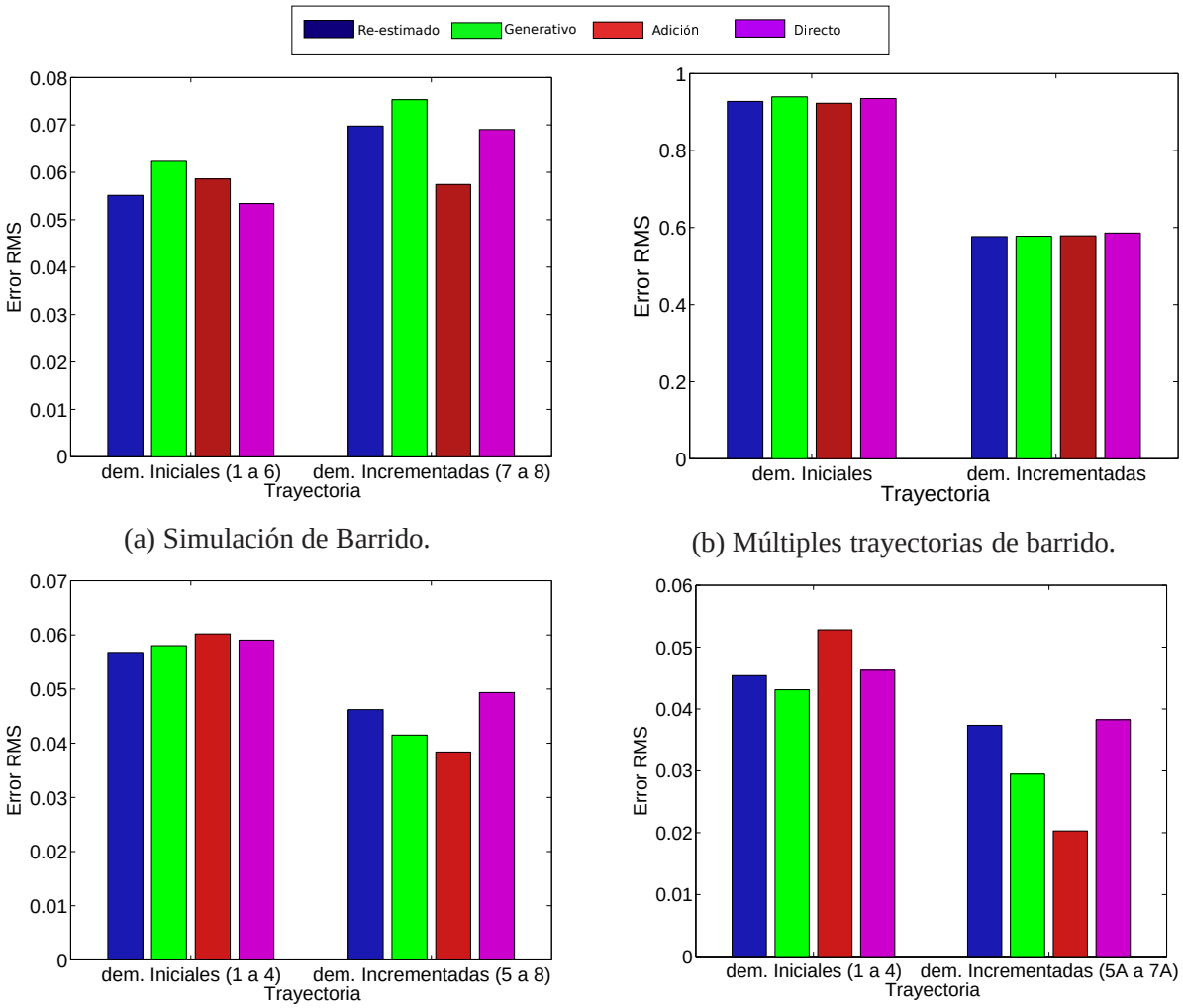


Figura 6-16: Error RMS para la tarea real usando las tres trayectorias de parámetros alejados. Azul: Estimado con las trayectorias de mano adelante. Verde: Generativa. Rojo: Adición de modelos.



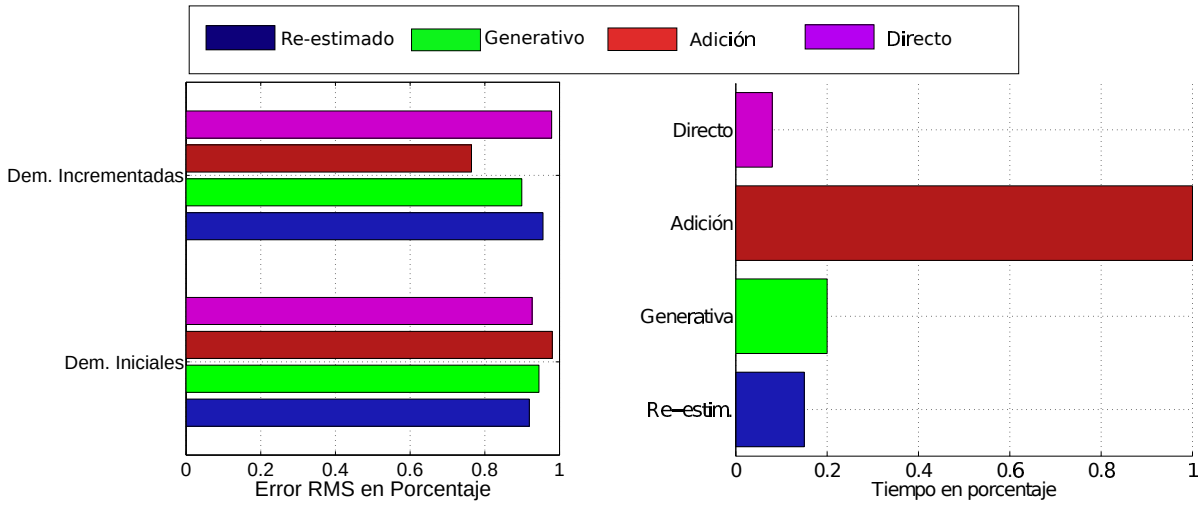
(a) Simulación de Barrido.

(b) Múltiples trayectorias de barrido.

(c) Tarea real de colocar la manga.

(d) Trayectorias con parámetros distantes.

Figura 6-17: Error RMS, para todas las pruebas y las tres técnicas propuestas. Se agruparon los resultados por tipo de trayectorias: iniciales e incrementadas. En azul con el modelo TPGMM re-estimado, en verde con la técnica generativa, en rojo con la técnica de adición de modelos, en magenta con la técnica de actualización directa.



(a) Errores RMS en porcentaje.

(b) Tiempos de Estimación en porcentaje.

Figura 6-18: Resultados generales de las técnicas propuestas. Los errores RMS y el tiempo son normalizados. En azul: Existente. En verde: Generativo. En rojo: Adición de modelos. En magenta: Actualización directa.

agrupan por tipo de trayectoria: iniciales e incrementadas. En la prueba de simulación de barrido, la técnica de adición de modelos obtiene los mejores resultados. En las pruebas de múltiples trayectorias y de colocar la manga, la técnica generativa iguala en resultados a la adición de modelos. En la prueba de trayectorias con parámetros más alejados, la técnica generativa obtiene los mejores resultados. En general (Figura **6-18**), los errores resultado de las técnicas propuestas son similares o ligeramente mayores que los del modelo TPGMM incrementado, aunque es de tener en cuenta que el modelo TPGMM incrementado se calcula directamente con las demostraciones iniciales e incrementadas. En el caso de las demostraciones iniciales, el mayor incremento del error RMS (Figura **6-18**a) se produce para la adición de modelos y es, aproximadamente, del $9\,\%$.

La técnica generativa tiene la ventaja de ser más rápida que la de adición de modelos, además el error es similar al obtenido con el modelo TPGMM estimado directamente con la información de las trayectorias. Su desventaja radica en que si se volviera a aplicar la técnica generativa, sobre un modelo resultado de un incremento con dicha técnica, el error se acumularía. Por lo anterior, aunque tarda más tiempo en la estimación, se considera que la técnica de adición de modelos es la mejor, ya que permite seguir incrementando trayectorias de manera continua.

Las trayectorias resultado del método directo tienden a alejarse de las demostradas, como se mostró en la Figura **6-7**. Lo anterior tal vez es debido a que en el caso de TPGMM, las trayectorias que se agregan tienen una probabilidad posterior diferente a las trayectorias existentes, lo cual es la base de la técnica de actualización directa.

El uso de las técnicas propuestas con más de una trayectoria mejora los resultados contra el caso donde solo se agrega una trayectoria.

6.4. Resumen

Se propusieron tres técnicas que permiten agregar trayectorias a un modelo TPGMM estimado previamente. La primera es la técnica generativa, la cual genera información de trayectorias a partir del modelo previo, las cuales se adicionan con las trayectorias a incrementar para obtener un nuevo modelo. La segunda es la técnica de adición de modelos, en la que se adicionan los parámetros del modelo previo con los parámetros del modelo obtenido de la o las nuevas trayectorias. La tercera es la técnica de actualización directa, que emplea una versión modificada del algoritmo de esperanza maximización, que permite incrementar el modelo usando las nuevas trayectorias y los parámetros del modelo previo. Las técnicas fueron probadas a través de simulaciones y una prueba real, el error RMS de los resultados mostró que las técnicas propuestas presentan mejores resultados que los que se obtienen cuando solo se emplea el modelo existente.

7 Conclusiones y Trabajo Futuro

A lo largo de esta tesis se propusieron soluciones a algunos problemas que surgen en el ámbito de generalización de trayectorias en PpD, tres situaciones tratadas fueron: *i)* Cómo generar nuevas trayectorias similares a las demostradas, tanto en el espacio de la tarea como en el de articulación, sin tener el modelo cinemático del robot?, *ii)* Cómo recuperarse de uno o varios casos de fallos de ejecución en tareas que requieren la manipulación de objetos deformables?, *iii)* Cómo aumentar el conocimiento de un modelo de generalización de trayectorias para que pueda generar, de manera adecuada, nuevas trayectorias?. Para cuantificar el desempeño de las técnicas propuestas se emplearon dos métricas, el error RMS entre la trayectoria obtenida con el modelo y la trayectoria demostrada; y el tiempo de estimación de la técnica.

Conclusiones

En el aprendizaje del modelo cinemático se empleó la red neuronal llamada máquina de aprendizaje extremo, esta es simple de implementar y presenta buena estabilidad, gracias al cálculo previo de los pesos y desplazamientos, como se expuso en el Capítulo 4. La estabilidad se comprobó experimentalmente mediante repeticiones de diversos experimentos. Para el aprendizaje de la cinemática se requiere el cálculo del jacobiano, el cual relaciona velocidades de articulación con velocidades del efector final del robot, el error del jacobiano es pequeño, al comparar el obtenido con la máquina de aprendizaje extremo contra el de la cinemática modelada. Otro tema importante es el tiempo de aprendizaje del modelo, el cual puede llegar a ser grande para una red neuronal de aprendizaje clásico (Ej. retro-propagación), dado que la cantidad de parejas (posición, articulación) es grande, al emplear la red propuesta este es de menos de 3 segundos, lo que permite su uso en aplicaciones reales.

Una de las aplicaciones realizadas en esta tesis fue el aprendizaje del modelo cinemático de un robot Katana de cinco grados de libertad, con este modelo aprendido, y una técnica de generalización de trayectorias. Se generaron trayectorias de la tarea de tomar y colocar, se constató que el sistema funciona de manera satisfactoria y que se puede obtener una nueva trayectoria cuando se presenta un cambio en la posición final del objeto a ubicar. La técnica presentada tiene dos restricciones, la primera es que la posición final se encuentre dentro de los puntos donde se entrenó la máquina de aprendizaje extremo que aprende la cinemática y, la segunda, es que no exceda de-

masiado la posición final de la trayectoria demostrada.

En tareas que involucren seres humanos y objetos deformables es necesario emplear técnicas de aprendizaje que sean tolerantes a cambios y a posibles fallos de ejecución. En este trabajo se propuso usar una combinación de: un árbol-kd con modelos de mezcla de gaussianas parametrizados en la tarea. El árbol permite la clasificación del fallo entre una serie de posibles fallos conocidos, los modelos de mezcla de gaussianas parametrizados en la tarea permiten actuar según el fallo clasificado. El sistema implementado permite que el robot coloque la manga de la camisa para diversas posiciones y curvas del brazo y que, además, pueda recuperarse ante un fallo en la ejecución de la tarea. Generalmente, la trayectoria directa se ejecuta correctamente la mayoría de las veces, pero en algunas ocasiones se pueden presentar enganches de la manga con el brazo, los cuales son detectados por un sensor de fuerza/torque. Es en este punto donde se ejecuta un modelo alternativo, el cual recupera el fallo de ejecución, logrando colocar la manga. Tanto la trayectoria directa como las que permiten la recuperación del fallo, fueron obtenidas usando programación por demostración.

La técnica propuesta de recuperación a fallos es modular en el sentido que permite agregar fácilmente nuevos casos de falla, para esto es necesario estimar un modelo de las trayectorias que recuperan el fallo y re-estimar el árbol-kd, agregándole una nuevo caso de falla. Se mostró que el sistema además de resolver el problema de enganche de la manga cerca al hombro, también permitió agregar un nuevo caso de fallo, que consistió en un enganche al inicio de la inserción de la manga, el sistema logró recuperarse a este caso.

Aumentar el conocimiento de un modelo de una tarea es la meta del aprendizaje incremental, en el caso de generalización de trayectorias con modelos de mezcla de gaussianas parametrizados en la tarea, se propusieron tres técnicas: *i)* Generativa, *ii)* Adición de modelos, *iii)* Actualización directa. Estas fueron probadas mediante la simulación de barrer un objeto y la tarea de colocar la manga de una camisa usando un brazo robótico, se puede observar en los resultados que al usar los modelos incrementados, la respuesta es mejor a la que se obtiene al usar solo el modelo existente.

En las pruebas realizadas las trayectorias agregadas para estimar el modelo incrementado fueron diferentes, en forma, a las que se tenían originalmente. Esto permite que el nuevo modelo tenga un comportamiento diferente, a expensas de un ligero incremento en el error, al que se tenía cuando se reproducían las trayectorias originales. Este incremento es de un $9\,\%$ en promedio para los experimentos realizados.

Las técnicas propuestas para aprendizaje incremental de modelos TPGMM presentaron resultados satisfactorios: reducción del error RMS en un $40\,\%$ en promedio con respecto a los resultados de usar el modelo existente; y errores similares o ligeramente mayores comparados con el obtenido

al realizar una re-estimación completa del modelo TPGMM. Aunque tarda más tiempo en la estimación, se considera que la técnica de adición de modelos es la mejor, ya que permite seguir incrementando de manera continua trayectorias. En segundo lugar está la técnica generativa, la cual presenta un compromiso aceptable entre el tiempo de estimación y los errores RMS de la trayectorias iniciales e incrementadas.

Trabajo Futuro

En el aprendizaje de la cinemática empleando la máquina de aprendizaje extremo se planea, a futuro, trabajar con un brazo robótico redundante y mirar el empleo de nuevas variantes de la máquina de aprendizaje extremo que resuelvan, de alguna manera, el problema de la selección de la cantidad de neuronas, como por ejemplo en [45].

En cuanto a la recuperación a fallos de ejecución, se planea considerar casos donde se presenten variaciones en tiempo de ejecución de las posiciones y curvaturas asociadas al movimiento del brazo. También se trabajará en la obtención de los parámetros de la tarea usando un sistema de visión.

En aprendizaje incremental se planea buscar técnicas para disminuir el tiempo que tarda la optimización, en la técnica de adición de modelos. También buscar otras técnicas que permitan adicionar al modelo existente nuevas trayectorias, basadas en el algoritmo de maximización de la esperanza. Además encontrar mejoras a los resultados de la técnica de actualización directa.

Referencias

[1] ALBU-SCHAFFER, A. ; HADDADIN, S. ; OTT, C. ; STEMMER, A. ; WIMBCK ; HIRZINGER, T.: The DLR lightweight robot - design and control concepts for robots in human environments. En: *Industrial Robot: An International Journal* 34 (2007), p. 376–385.

[2] ALEOTTI, Jacopo ; CASELLI, Stefano ; REGGIANI, Monica: Toward Programming of Assembly Tasks by Demonstration in Virtual Environments, Robot and Human Interactive. En: *Communication*, 2003

[3] ALISSANDRAKIS, Aris ; NEHANIV, Chrystopher L. ; DAUTENHAHN, Kerstin: Action, State and Effect Metrics for Robot Imitation. En: *15th IEEE International Symposium on Robot and Human Interactive Communication*. Hatfield, UK, 2006

[4] ALIZADEH, Tohid ; CALINON, Sylvain ; CALDWELL, Darwin: Learning from demonstrations with partially observable task parameters. En: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014, p. 3309–3314

[5] AMIT, R. ; MATARIC, Maja J.: A Correspondence Metric for Imitation. En: *AAAI*, 2004, p. 944–945

[6] ARGALL, Brenna D. ; CHERNOVA, Sonia ; VELOSO, Manuela ; BROWNING, Brett: A survey of robot learning from demonstration. En: *Robotics and Autonomous Systems* 57 (2009), p. 469–483

[7] ARISTIDOU, Andreas ; LASENBY, Joan: Inverse Kinematics: a review of existing techniques and introduction of a new fast iterative solver. En: *University of Cambridge, Technical Report*, 2009

[8] ASADA, Haruhiko ; LIU, Sheng: Transfer of Human Skills to Neural Net Robot Controllers. En: *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, 1991

[9] AZAD, Pedram ; UDE, Ales ; ASFOUR, Tamim ; CHENG, Gordon ; DILLMANN, Ruediger: Image-Based Markerless 3d Human Motion Capture using Multiple Cues. En: *HRI*, 2006

[10] BENTIVEGNA, Darrin C. ; ATKESON, Christopher G.: A Framework for Learning from Observation Using Primitives. En: KAMINKA, GalA. (Ed.) ; LIMA, PedroU. (Ed.) ; ROJAS, Raúl (Ed.): *RoboCup 2002: Robot Soccer World Cup VI* Vol. 2752. Springer Berlin Heidelberg, 2003. – ISBN 978–3–540–40666–2, p. 263–270

[11] BENTLEY, Jon L.: Multidimensional Binary Search Trees Used for Associative Searching. En: *Communications of the ACM* 18 (1975), p. 509–517

[12] BIGGS, Geoffrey ; MACDONALD, Bruce: A Survey of Robot Programming Systems. En: *in Proceedings of the Australasian Conference on Robotics and Automation, CSIRO*, 2003, p. 27

[13] BILLARD, A. ; HAYES, G.: DRAMA a Connectionist Architecture for Control and Learning in Autonomous Robots. En: *Adaptive behavior*, 1999, p. 35–63

[14] BILLARD, Aude ; CALINON, Sylvain ; DILLMANN, Ruediger ; STEFAN, Schaal: Robot Programming by Demonstration. En: BRUNO SICILIANO, Oussama K. (Ed.): *Handbook of Robotics*. Berlin : Springer, 2008, Kapitel 59, p. 1371–1394

[15] BILLARD, Aude G. *MACHINE LEARNING TECHNIQUES*. Ecole Polytechnique Federale de Laussane. 2011

[16] BILLARD, Aude G. ; CALINON, Sylvain ; GUENTER, Florent: Discriminative and adaptive imitation in uni-manual and bi-manual tasks. En: *Robotics and Autonomous Systems, Elsevier* 54 (2006), p. 370–384

[17] BREAZEAL, Cynthia ; SCASSELLATI, Brian: Robots that imitate humans. En: *trends in Cognitive Sciences*, 2002

[18] CALINON, S. ; BILLARD, A.: Stochastic Gesture Production and Recognition Model for a Humanoid Robot. En: *proceedings of IEEE international conference on intelligent robots and systems*, 2004, p. 2769–2774

[19] CALINON, S. ; D'HALLUIN, F. ; CALDWELL, D.G. ; BILLARD, A.G.: Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework. En: *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on*, 2009, p. 582–588

[20] CALINON, S. ; DHALLUIN, F. ; SAUSER, E. ; CALDWELL, D. ; BILLARD, A.: Learning and Reproduction of Gestures by Imitation. En: *IEEE robotics and automation magazine*, 2010, p. 44–54

[21] CALINON, S. ; GUENTER, F. ; BILLARD, A.: On Learning, Representing and Generalizing a Task in a Humanoid Robot. En: *IEEE Transactions on Systems, Man and Cybernetics, Part B.*, 2007, p. 286–298

[22] CALINON, S. ; SARDELLITTI, I. ; CALDWELL, D. G.: Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies. En: *Proc. IEEE/RSJ Intl Conf. on Intelligent Robots and Systems (IROS)*, 2010

[23] CALINON, Sylvain ; ALIZADEH, Tohid ; CALDWELL, Darwin: On Improving the Extrapolation Capability of Task-Parameterized Movement Models. En: *IEEE International Conference on Intelligent Robots and Systems*, 2013

[24] CALINON, Sylvain ; BILLARD, Aude: Incremental Learning of Gestures by Imitation in a Humanoid Robot. En: *IEEE Human Robot Interation Conference*, 2007

[25] CALINON, Sylvain ; BILLARD, Aude: Statistical Learning by Imitation of Competing Constraints in Joint Space and Task Space. En: *Advanced Robotics*, 23 (2009), Nr. 15, p. 2059–2076

[26] CALINON, Sylvain ; GUENTER, Florent ; BILLARD, Aude: Goal-Directed Imitation in a Humanoid Robot. En: *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2004

[27] CALINON, Sylvain ; GUENTER, Florent ; BILLARD, Aude: On Learning, Representing and Generalizing a Task in a Humanoid Robot. En: *IEEE Transactions on Systems, Man and Cybernetics, part b*, 2007

[28] CALINON, Sylvain ; ZHIBIN LI, and Nikos G. T. ; CALDWELL, Darwin: Statistical dynamical systems for skills acquisition in humanoids. En: *8th IEEE-RAS International Conference on Humanoid Robots*, 2012

[29] CEDERBORG, Thomas ; LI, Ming ; BARANES, Adrien ; OUDEYER, Pierre-Yves: Incremental Local Online Gaussian Mixture Regression for Imitation Learning of Multiple Tasks. En: *IEEE International Conference on Intelligent Robots and Systems*, 2010

[30] CHACKO, B. ; ANTO, B.: Online Sequential Extreme Learning Machine Based Handwritten Character Recognition. En: *Proceeding of the 2011 IEEE Students Technology Symposium*, 2011, p. 142–147

[31] CHAMINADE, Thierry ; OZTOP, Erhan ; CHENG, Gordon ; KAWATO, Mitsuo: From self-observation to imitation: Visuomotor association on a robotic hand. En: *Brain Research Bulletin 75, Elsevier*, 2008

[32] CHANG, G. ; KULIC, Dana: Robot Task Error Recovery Using Petri Nets Learned from Demonstration. En: *International Conference on Advanced Robotics (ICAR),*, 2013, p. 1–6

[33] CHANG, Guoting ; KULIC, D.: Motion learning from observation using Affinity Propagation clustering. En: *RO-MAN, 2013 IEEE*, 2013, p. 662–667

[34] CHUNG, Michael ; FORBES, Maxwell ; CAKMAK, Maya ; RAO, Rajesh P. N.: Accelerating Imitation Learning through Crowdsourcing. En: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014

[35] COLOMÉ, Adriá ; NEUMANN, Gerhard ; PETERS, Jan ; TORRAS, Carme: Dimensionality Reduction for Probabilistic Movement Primitives. En: *IEEE International congress of Robotics and automation*, 2014

[36] COOPER, G. ; HERSKOVITS, E.: A Bayesian Method for the Induction of Probabilistic Networks from Data. En: *Machine Learning, Kluwer Academic Publishers* 9 (1992), p. 309–347

[37] CORKE, Peter I.: Robotics, Vision & Control: Fundamental Algorithms in Matlab. En: *Springer*, 2011

[38] CRAIG, J. ; PEARSON (Ed.): *Introduction to Robotics - Mechanics and Control*. Third edition. Pearson, 2005

[39] DANTAM, N. ; ESSA, I ; STILMAN, M.: Linguistic transfer of human assembly tasks to robots. En: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, p. 237–242

[40] DENG, W. ; ZHENG, Q. ; CHEN, L.: Regularized Extreme Learning Machine. En: *IEEE Symposium on Computational Intelligence and Data Mining*, 2009, p. 389 – 395

[41] DILLMANN, R. ; ROGALLA, O. ; EHRENMANN, M. ; ZOLLNER, R. ; BORDEGONI, M.: Learning Robot Behaviour and Skills Based on Human Demonstration and Advice: The Machine Learning Paradigm. En: *Citeseer*, 1999

[42] D'SOUZA, A. ; VIJAYAKUMAR, S. ; SCHAAL, S.: Learning inverse kinematics. En: *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on* Vol. 1, 2001, p. 298–303 vol.1

[43] EKVALL, S. ; KRAGIC, D.: Learning Task Models from Multiple Human Demonstrations. En: *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, 2006, p. 358–363

[44] ERLHAGEN, Wolfram ; MUKOVSKIY, Albert ; BICHO, Estela ; PANIN, Giorgio ; KISS, Csaba ; KNOLL, Alois ; VAN SCHIE, Hein ; BEKKERING, Ha r.: Action Understanding and Imitation Learning in a Robot-Human Task. En: DUCH, Wlodzislaw (Ed.) ; KACPRZYK, Janusz (Ed.) ; OJA, Erkki (Ed.) ; ZADROZNY, Slawomir (Ed.): *Artificial Neural Networks: Biological Inspirations – ICANN 2005* Vol. 3696. Springer Berlin Heidelberg, 2005, p. 261–268

[45] FENG, Guorui ; HUANG, Guang-Bin ; LIN, Qingping ; GAY, R.: Error Minimized Extreme Learning Machine With Growth of Hidden Nodes and Incremental Learning. En: *Neural Networks, IEEE Transactions on* 20 (2009), Aug, Nr. 8, p. 1352–1357

[46] FLASH, T. ; HOGAN, N.: The Coordination of Arm Movements: An Experimentally Confirmed Mathematical Model. En: *Journal of Neuroscience, Society for Neuroscience*, 1985, p. 1688–1703

[47] FOD, Ajo ; MATARIC, Maja J. ; JENKINS, Odest C.: Automated Derivation of Primitives for Movement Classification. En: *Autonomous Robots* 12 (2002), Nr. 1, p. 39–54

[48] FORTE, D. ; UDE, A. ; GAMS, A.: Real-time generalization and integration of different movement primitives. En: *11th IEEE-RAS International Conference on Humanoid Robots (Humanoids),*, 2011, p. 590–595

[49] GARCIA, Elena ; JIMENEZ, Maria ; DE SANTOS, Pablo G. ; ARMADA, Manuel: The Evolution of Robotics Research - From Industrial Robotics to Field and Service Robotics. En: *IEEE Robotics and Automation Magazine*, 2007

[50] GINI, Maria ; GINI, Giuseppina: Towards automatic error recovery in robot programs. En: *Lecture Notes in Computer Science, Springer,* 168 (1984), p. 411–416

[51] GRIMES, David B. ; RAWICHOTE, Chalodhorn ; RAO, Rajesh P. N.: Dynamic Imitation in a Humanoid Robot through Nonparametric Probabilistic Inference. En: *Robotics, Science and Systems*, 2006

[52] GUENTER, F. ; BILLARD, A.G.: Using reinforcement learning to adapt an imitation task. En: *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, 2007, p. 1022–1027

[53] GUENTER, F. ; HERSCH, M. ; CALINON, S. ; BILLARD, A.: Reinforcement Learning for Imitating Constrained Reaching Movements. En: *Advanced Robotics, Special Issue on Imitative Robots* 21 (2007), Nr. 13, p. 1521–1544

[54] HALL, Peter ; HICKS, Yulia: A Method to add Gaussian Mixture Models. En: *Technical Report,University of Bath*, 2004

[55] HASAN, Ali T. ; ISMAIL, N. ; HAMOUDA, A.M.S. ; ARIS, Ishak ; MARHABAN, M.H. ; AL-ASSADI, H.M.A.A.: Artificial neural network-based kinematics Jacobian solution for serial manipulator passing through singular configurations. En: *Advances in Engineering Software 41, Elsevier*, 2010, p. 359–367

[56] HERSCH, Micha ; GUENTER, Florent ; CALINON, Sylvain ; BILLARD, Aude: Learning Dynamical System Modulation for Constrained Reaching Tasks. En: *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, 2006, p. 444–449

[57] HERZOG, Dennis ; KRUGER, Volker ; GREST, Daniel: Parametric Hidden Markov Models for Recognition and Synthesis of Movements. En: *British Machine Vision Conference*, 2008

[58] HERZOG, Dennis ; UDE, Ales ; KRUGER, Volker: Motion Imitation and Recognition using Parametric Hidden Markov Models. En: *8th IEEE-RAS International Conference on Humanoid Robots*, 2008

[59] HOANG, M. ; HUYNH, H. ; VO, N. ; WON, Y.: A Robust Online Sequential Extreme Learning Machine. En: *Advances in Neural Networks, Springer*, 2007, p. 1077–1086

[60] HOYOS-GUTIÉRREZ, Jose ; PEÑA-SOLÓRZANO, Carlos ; GARZÓN-CASTRO, Claudia ; PRIETO-ORTIZ, Flavio: Hacia el Manejo de una Herramienta por un Robot NAO Usando Programación por Demostración. En: *revista tecnológicas, ITM, Colombia* 17 (2014), Nr. 33, p. 65–76

[61] HUANG, G-B ; ZHU, Q-Y ; SIEW, C-K: Extreme learning machine: a new learning scheme of feedforward neural networks. En: *Proceedings of international joint conference on neural networks (IJCNN2004)*, 2004, p. 985[U+0096]990

[62] HUANG, G-B ; ZHU, Q-Y ; SIEW, C-K: Extreme learning machine theory and applications. En: *Neurocomputing N. 70, Elsevier*, 2006, p. 489–501

[63] IJSPEERT, Auke J. ; NAKANISHI, Jun ; STEFAN, Schaal: Trajectory Formation for Imitation with Nonlinear Dynamical Systems. En: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001

[64] INAMURA, T. ; NAKAMURA, Y. ; SHIMOZAKI, M.: Associative Computational Model of Mirror Neurons that connects Missing Link between Behaviors and Symbols. En: *Proceedings of the 2002 IEEE/RSJ Int. conf. on intelligent robots and systems*, 2002, p. 1032–1037

[65] INAMURA, Tetsunari ; TOSHIMA, Iwaki ; TANIE, Hiroaki ; NAKAMURA, Yoshihiko: Embodied Symbol Emergence Based on Mimesis Theory. En: *The International Journal of Robotics Research, Sage pub.*, 2004

[66] ITO, M. ; TANI, J.: On-line Imitative Interaction with a Humanoid Robot Using a Dynamic Neural Network Model of a Mirror System. En: *Adaptive behavior*, 2004, p. 93–115

[67] KADONE, H. ; NAKAMURA, Y.: Segmentation, Memorization, Recognition and Abstraction of Humanoid Motions Based on Correlations and Associative Memory. En: *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, 2006, p. 1–6

[68] KAISER, M. ; DILLMANN, R.: Building Elementary Robot Skills from Human Demonstration. En: *Proceedings of IEEE lnternational Conference on Robotics and Automation*, 1996, p. 2700–2705

[69] KIM, Seungsu ; GRIBOVSKAYA, E. ; BILLARD, A.: Learning motion dynamics to catch a moving object. En: *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, 2010, p. 106–111

[70] KOBER, J. ; PETERS, J.: Imitation and Reinforcement Learning. En: *Robotics Automation Magazine, IEEE* 17 (2010), Nr. 2, p. 55–62

[71] KOBER, Jens ; MOHLER, Betty ; PETERS, Jan: Imitation and Reinforcement Learning for Motor Primitives with Perceptual Coupling. En: *Studies in Computational Intelligence, Springer*, 2010

[72] KRAGIC, D. ; SONG, D. ; HUEBNER, K. ; KYRKI, V.: Learning Task Constraints for Robot Grasping using Graphical Models. En: *IROS*, 2010

[73] KRUG, R. ; DIMITROVZ, D.: Representing movement primitives as implicit dynamical systems learned from multiple demonstrations. En: *Advanced Robotics (ICAR), 2013 16th International Conference on*, 2013, p. 1–8

[74] KRUGER, V. ; HERZOG, D. ; BABY, S. ; UDE, A ; KRAGIC, D.: Learning Actions from Observations. En: *Robotics Automation Magazine, IEEE* 17 (2010), June, Nr. 2, p. 30–43

[75] KULIC, Dana ; TAKANO, Wataru ; NAKAMURA, Yoshihiko: Incremental Learning, Clustering and Hierarchy Formation of Whole Body Motion Patterns using Adaptive Hidden Markov Chains. En: *The International Journal of Robotics Research* 27 (2008), Nr. 7, p. 761 – 784

[76] KUNIYOSHI, Yasuo ; YOROZU, Yasuaki ; INABA, Masayuki ; INOUE, Hirochika: From Visuo-Motor Self Learning to Early Imitation - A Neural Architecture for Humanoid Learning. En: *Proceedings of Ihe IEEE International Conference on Robotics and Automation*, 2003

[77] LAMBRECHT, Jens ; KLEINSORGE, Martin ; ROSENSTRAUCH, Martin ; KRUGER, Jorg: Spatial Programming for Industrial Robots Through Task Demonstration. En: *Int J Adv Robot Syst*, 10:254. (2013,), p. 1–10

[78] LEE, Dongheui ; OTT, Christian: Incremental kinesthetic teaching of motion primitives using the motion refinement tube. En: *Autonomous Robot*, 2011, p. 115–131

[79] LEE, George ; LIN, Hsien-I: Understanding Robot Motor Capability using Information-Theory-Based Approach. En: *IEEE/RSJ International Conference on Intelligent Robots and Systems, USA*, 2009

[80] LEE, Kyuhwa ; DEMIRIS, Yiannis: Towards Incremental Learning of Task-dependent Action Sequences using Probabilistic Parsing. En: *IEEE First Joint International Conference on Development and Learning and on Epigenetic Robotics (ICDL-EPIROB 2011)*, 2011

[81] LEE, Sang H. ; KIM, Hyung K. ; SUH, Il-Hong: Incremental learning of primitive skills from demonstration of a task. En: *6th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2011, p. 185–186

[82] LEON, Adrian ; MORALES, Eduardo F. ; ALTAMIRANO, Leopoldo ; RUIZ, Jaime R.: Teaching a robot to perform task through imitation and on-line feedback. En: *In Proc. Workshop: New Developments in Imitation Learning, as part of the Internacional Conference on Machine Learning*, 2011

[83] LI, G. ; NIU, P.: An enhanced extreme learning machine based on ridge regression for regression. En: *Neural Comput & Applications, Springer*, 2011, p. 1–8

[84] LIN, J.F.-S. ; KULIC, D.: Segmenting human motion for automated rehabilitation exercise analysis. En: *Engineering in Medicine and Biology Society (EMBC), 2012 Annual International Conference of the IEEE*, 2012, p. 2881–2884

[85] LOPES, Manuel ; CABIDO, Fernando: A Developmental Roadmap for Learning by Imitation in Robots. En: *Universidade Técnica de Lisboa, Instituto Superior Técnico, Tesis doctoral*, 2006

[86] LOPES, Manuel ; MONTESANO, Luis ; BERNARDINO, Alexandre ; SANTOS VICTOR, Jose: Learning Object Affordances From Sensory Motor Coordination to Imitation. En: *IEEE Trans. on robotics*, 2008

[87] LOPES, Manuel ; SANTOS-VICTOR, Jose: A Developmental Roadmap for Learning by Imitation in Robots. En: *IEEE Transactions on Systems, Man, and Cybernetics*, 2007

[88] MARTÍNEZ, David ; ALENYÀ, Guillem ; JIMÉNEZ, Pablo ; TORRAS, Carme ; ROSSMANN, Jürgen ; WANTIA, Nils ; AKSOY, Eren E. ; HALLER, Simon ; PIATER, Justus: Active Learning of Manipulation Sequences. En: *IEEE International Conference on Robotics and Automation (ICRA)*, 2014

[89] MATSUBARA, Takamitsu ; HYON, Sang-Ho ; MORIMOTO, Jun: Learning Stylistic Dynamic Movement Primitives from Multiple Demonstrations. En: *IROS*, 2010

[90] MEIER, Franziska ; THEODOROU, E. ; STULP, F. ; SCHAAL, S.: Movement segmentation using a primitive library. En: *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, 2011, p. 3407–3412

[91] MERICLI, C. ; VELOSO, M.: Biped Walk Learning Through Playback and Corrective Demonstration. En: *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI'10),*, 2010, p. 1–6

[92] MERICLI, C. ; VELOSO, M. ; AKIN, H.L.: Efficient task execution and refinement through multi-resolution corrective demonstration. En: *IEEE International Conference on Robotics and Automation (ICRA),*, 2012, p. 1805–1810

[93] MERICLI, C. ; VELOSO, M. ; H.L., Akin: Task Refinement for Autonomous Robots using Complementary Corrective Human Feedback. En: *International Journal of Advanced Robotic Systems, Intech Open Science* 8 (2011), p. 68–79

[94] MINHAS, Rashid ; BARADARANI, Aryaz ; SEIFZADEH, Sepideh ; WU, Q.M. J.: Human action recognition using extreme learning machine based on visual vocabularies. En: *Neurocomputing, Elsevier*, 2010, p. 1906–1917

[95] MIYAMOTO, Hiroyuki ; SCHAAL, Stefan ; KAWATO, Mitsuo et a.: A Kendama Learning Robot Based on Bi-directional Theory. En: *Neural Networks*, Elsevier, 1996

[96] MUHLIG, M. ; GIENGER, M. ; HELLBACH, Sven ; STEIL, J.J. ; GOERICK, C.: Task-level imitation learning using variance-based movement optimization. En: *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, 2009, p. 1177–1184

[97] MURPHY, K.: Hidden Markov Model (HMM) Toolbox for Matlab. En: *http://www.cs.ubc.ca/ murphyk/Software/HMM/hmm.html*, 2012

[98] NEHANIV, Chrystopher L. ; DAUTENHAHN, Kerstin: Like me?- Measures of Correspondence and Imitation. En: *Cybernetics and Systems: An International Journal*, 2001

[99] NEMEC, B. ; GAMS, A. ; UDE, A.: Velocity Adaptation for Self-Improvement of Skills Learned from User Demonstrations. En: *13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2013, p. pp. 423–428

[100] NIEKUM, Scott ; CHITTA, Sachin ; MARTHI, Bhaskara ; OSENTOSKI, Sarah ; BARTO, Andrew G.: Incremental Semantically Grounded Learning from Demonstration. En: *Robotics: Science and Systems*, 2013

[101] NOPE, Sandra ; LOAIZA, Humberto ; CAICEDO, Eduardo: Imitación de gestos por brazos robóticos: Una propuesta para evaluar su calidad. En: *DYNA Ingeniería e Industria, Vol. 85 no 5, España*, 2010

[102] NOPE, Sandra ; LOAIZA, Humberto ; CAICEDO, Eduardo: Programación de un robot bajo el paradigma del aprendizaje por demostración. En: *Rev. Fac. Ing. Univ. Antioquia N. 58*, 2011

[103] OZTOP, Erhan ; KAWATO, Mitsuo ; ARBIB, Michael: Mirror neurons and imitation: A computationally guided review. En: *Neural Networks*, Elsevier, 2006

[104] PAIS, L. ; BILLARD, A. *Tactile interface user-friendliness evaluated in the context of robot programming by demonstration*. HRI Workshop on Advances in tactile sensing and touch based human-robot interaction. 2012

[105] PARDOWITZ, M. ; KNOOP, S. ; DILLMANN, R. ; ZOLLNER, R.D.: Incremental Learning of Tasks From User Demonstrations, Past Experiences, and Vocal Comments. En: *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 37 (2007), April, Nr. 2, p. 322–332

[106] PARK, Dae-Hyung ; HOFFMANN, H. ; PASTOR, Peter ; SCHAAL, S.: Movement reproduction and obstacle avoidance with dynamic movement primitives and potential fields. En: *8th IEEE-RAS International Conference on Humanoid Robots,*, 2008, p. 91–98

[107] PASTOR, Peter ; HOFFMANN, H. ; ASFOUR, T. ; SCHAAL, S.: Learning and generalization of motor skills by learning from demonstration. En: *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, 2009, p. 763–768

[108] PHUNG, A. S. ; MALZAHN, J. ; HOFFMANN, F. ; BERTRAM, T.: Get Out of the Way - Obstacle Avoidance and Learning by Demonstration for Manipulation. En: *IFAC World Congress*, 2011

[109] PIRES, J. N. ; VEIGA, Germano ; ARAÚJO, Ricardo: Programming-by-demonstration in the coworker scenario for SMEs. En: *Journal of Industrial Robot* 36 (2009), Nr. 1, p. 73–83

[110] POMPLUN, M. ; MATARIC, M.J.: Evaluation Metrics and Results of Human Arm Movement Imitation. En: *Proceedings of the First IEEE-RAS International Conference on Humanoid Robots, Humanoids*, 2000

[111] RAMBOW, M. ; SCHAUSS, T. ; BUSS, M. ; HIRCHE, S.: Autonomous manipulation of deformable objects based on teleoperated demonstrations. En: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, p. 2809–2814

[112] RAMISA, A. ; ALENYA, G. ; MORENO-NOGUER, F. ; TORRAS, C.: Using depth and appearance features for informed robot grasping of highly wrinkled clothes. En: *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, p. 1703–1708

[113] REMPIS, C.W.: A neural network to capture demonstrated motions on a humanoid robot to rapidly create complex central pattern generators as reusable neural building blocks. En: *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, 2013, p. 5291–5296

[114] ROZO, L. ; CALINON, S. ; CALDWELL, D. G. ; JIMENEZ, P. ; TORRAS, C.: Learning collaborative impedance-based robot behaviors. En: *Conference on Artificial Intelligence (AAAI)*, 2013, p. 1422–1428

[115] ROZO, L. ; CALINON, S. ; CALDWELL, D.G.: Learning force and position constraints in human-robot cooperative transportation. En: *Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on*, 2014, p. 619–624

[116] ROZO, L. ; JIMENEZ, P. ; TORRAS, C.: Force-based robot learning of pouring skills using parametric hidden Markov models. En: *Robot Motion and Control (RoMoCo), 2013 9th Workshop on*, 2013, p. 227–232

[117] SCHAAL, Stefan: Is Imitation Learning the Route to Humanoid Robots? En: *Trends in Cognitive Sciences*, 1999

[118] SCHAAL, Stefan ; PETERS, Jan ; NAKANISHI, Jun ; IJSPEERT, Auke: Control, Planning, Learning, and Imitation with Dynamic Movement Primitives. En: *IEEE International Conference on Intelligent Robots and Systems*, 2003

[119] SCHAAL, Stefan ; PETERS, Jan ; NAKANISHI, Jun ; IJSPEERT, Auke: Learning movement primitives, International Symposium on Robotics Research (ISRR2003). En: *Springer Tracts in Advanced Robotics*, 2004

[120] SHINOHARA, D. ; MATSUBARA, T. ; KIDODE, M.: Learning motor skills with non-rigid materials by reinforcement learning. En: *IEEE International Conference on Robotics and Biomimetics (ROBIO),*, 2011, p. 2676–2681

[121] SHON, Aaron P. ; STORZ, Joshua J. ; RAO, Rajesh P. N.: Towards a Real-Time Bayesian Imitation System for a Humanoid Robot. En: *IEEE International Conference on Robotics and Automation*, 2007

[122] SKOGLUND, A. ; ILIEV, B. ; KADMIRY, B. ; PALM, R.: Programming by Demonstration of Pick-and-Place Tasks for Industrial Manipulators using Task Primitives. En: *Computational Intelligence in Robotics and Automation, 2007. CIRA 2007. International Symposium on*, 2007, p. 368–373

[123] SKOGLUND, A. ; ILIEV, B. ; PALM, R.: A Hand State Approach to Imitation with a Next-State-Planner for Industrial Manipulators. En: *Proceedings of the 2008 International Conference onCognitive Systems*, 2008, p. 130–137

[124] SOLE, M. ; TSOEU, M.S.: Sign Language Recognition using the Extreme Learning Machine. En: *IEEE Africon 2011*, 2011, p. 1–6

[125] SONG, Dan ; EK, C.H. ; HUEBNER, K. ; KRAGIC, D.: Multivariate discretization for Bayesian Network structure learning in robot grasping. En: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011, p. 1944–1950

[126] STIPANCIC, T. ; JERBIC, B. ; BUCEVIC, A. ; CURKOVIC, P.: PROGRAMMING AN INDUSTRIAL ROBOT BY DEMONSTRATION. En: *Proceedings of the 23rd International DAAAM Symposium*, 2012

[127] STULP, Freek ; RAIOLA, Gennaro ; HOARAU, Antoine ; IVALDI, Serena ; SIGAUD, Olivier: Learning Compact Parameterized Skills with a Single Regression. En: *IEEE-RAS International Conference on Humanoid Robots*, 2013

[128] SULEMAN, Muhammad ; AWAIS, Mian M.: Learning from demonstration in robots Experimental comparison of neural architectures. En: *Robotics and Computer-Integrated Manufacturing*, Elsevier, 2011

[129] TAMEI, T. ; MATSUBARA, T. ; RAI, A. ; SHIBATA, T.: Reinforcement learning of clothing assistance with a dual-arm robot. En: *11th IEEE-RAS International Conference on Humanoid Robots,*, 2011, p. 733–738

[130] TENORIO-GONZALEZ, A.C. ; MORALES, E.F. ; VILLASEÑOR-PINEDA, L.: Dynamic Reward Shaping: Training a robot by voice. En: *Proceedings of the Ibero–American Conference on Artificial Intelligence (IBERAMIA- 2010), Lecture Notes in Artificial Intelligence (LNAI-6433)*, Springer-Verlag., 2010, p. 483–492 ,

[131] TIKHANOFF, V. ; FITZPATRICK, P. ; ET AL: The iCub Humanoid Robot Simulator. En: *IROS Workshop on Robot Simulators*, 2008, p. 1–2

[132] TOWNSEND, William T. ; GUERTIN, Jeffrey A.: Teleoperator slave –WAM design methodology. En: *Industrial Robot* 26 (1999), Nr. 3, p. 167–177

[133] UDE, A. ; GAMS, A. ; ASFOUR, T. ; MORIMOTO, Jun: Task-Specific Generalization of Discrete and Periodic Dynamic Movement Primitives. En: *Robotics, IEEE Transactions on* 26 (2010), Nr. 5, p. 800–815

[134] WILSON, AD. ; BOBICK, AF.: Parametric hidden Markov models for gesture recognition. En: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 21 (1999), Sep, Nr. 9, p. 884–900

[135] WU, Ying ; HUANG, Thomas S.: Capturing articulated human hand motion: A divide and conquer approach. En: *IEEE conf. on computer vision*, 1999

[136] Y, Feng ; YAO-NAN, Wang ; YI-MIN, Yang: Inverse Kinematics Solution for Robot Manipulator based on Neural Network under Joint Subspace. En: *International Journal of Computer Communications and control, University of Oradea* 7 (2012), p. 459–472

[137] YUAN, X.: An interactive approach of assembly planning. En: *Computer Science Publications, University of Windsor* 1 (2002), p. 1–9

[138] ZUHER, F. ; ROMERO, R.: Recognition of Human Motions for Imitation and Control of a Humanoid Robot. En: *Robotics Symposium and Latin American Robotics Symposium (SBR-LARS), 2012 Brazilian*, 2012, p. 190–195