Tecno Lógicas
ISSN 0123-7799
Vol. 17, No. 33, pp. 65-76
Julio-diciembre de 2014

Hacia el manejo de una herramienta por un robot NAO usando programación por demostración

Towards tool handling by a NAO robot using programming by demonstration

José G. Hoyos-Gutiérrez[1], Carlos A. Peña-Solórzano[2], Claudia L. Garzón-Castro[3], Flavio A. Prieto-Ortiz[4] y Jorge G. Ayala-Garzón[5]

Recibido: 03 de junio de 2014,
Aceptado: 30 de junio de 2014

Cómo citar / How to cite
J. G. Hoyos-Gutiérrez, C. A. Peña-Solórzano, C. L. Garzón-Castro, F. A. Prieto-Ortiz y J. G. Ayala-Garzón, "Hacia el manejo de una herramienta por un robot NAO usando programación por demostración", *Tecno Lógicas*, vol. 17, no. 33, pp. 65-76, 2014.

1 Magíster en Ingeniería Eléctrica, Programa de Tecnología en Instrumentación Electrónica, Universidad del Quindío, Armenia-Colombia, josegabrielh@uniquindio.edu.co
2 Ingeniero Electrónico, Est. Maestría en Automatización Industrial, Universidad Nacional de Colombia, Bogotá-Colombia, capenas@unal.edu.co
3 Magíster en Automatización Industrial, Facultad de Ingeniería, Universidad de La Sabana, Chía-Colombia, claudia.garzon@unisabana.edu.co
4 Doctor en Automática, Facultad de Ingeniería, Universidad Nacional de Colombia, Bogotá-Colombia, faprieto@unal.edu.co
5 Estudiante de Ingeniería Informática, Universidad de La Sabana, Bogotá-Colombia, jorgeayga@unisabana.edu.co

Resumen

La programación por demostración es una técnica, donde, al contrario de la programación detallada que convencionalmente se hace al robot, este aprende a partir de una o varias demostraciones por parte de un humano u otro robot. Se presenta una técnica que permite a un robot humanoide, llevar una herramienta hasta la cabeza de un tornillo, siguiendo una trayectoria similar a las trayectorias demostradas por un ser humano. Además, permite variaciones en la ubicación y orientación de la cabeza del tornillo. Esto se logra gracias al uso de técnicas de visión de máquina y de modelos probabilísticos estimados a partir de las trayectorias de las demostraciones. El procesamiento de la imagen consistió en la segmentación en el espacio de color, la selección de puntos de interés de la cabeza del tornillo y a partir de estos el cálculo de su posición y orientación. La trayectoria que se requiere ante las variaciones, se genera usando la técnica de modelo de mezclas de Gaussianas parametrizado en la tarea. A través de gráficas de las nuevas trayectorias e imágenes de la secuencia del robot ejecutando estas trayectorias, se muestra el funcionamiento de la técnica. Pese a algunas limitaciones de la plataforma robótica utilizada, se lograron resultados aceptables.

Palabras clave

Programación por demostración, visión de máquina, modelos probabilísticos, humanoides, manejo de herramientas por robots.

Abstract

Programming by demonstration is a technique where, contrary to detailed programming, the robot learns from one or several demonstrations of the execution of the task by a human or another robot. A technique which allows a humanoid robot, take a tool to the head of a screw, following a similar trajectory than demonstrated trajectory by a human being, is presented. The technique also allows variations in location and orientation in the screw head. This is achieved thanks to the use of machine vision techniques and probabilistic models estimated from the demonstrated trajectories. Image processing consists of color space segmentation, and interest point's selection in the screw head to calculate its position and orientation. The required trajectory in the presence of variations is generated using parameterized Gaussian mixture models in the task. With the above, the new required trajectories are generated in accordance to the variations. The system operation is presented through the illustration of the new trajectories, and pictures of the robot following these trajectories. Despite some limitations of the robotic platform used, acceptable results were achieved.

Keywords

Programming by demonstration, machine vision, probabilistic models, humanoids, tool use in robots.

1. INTRODUCCIÓN

La programación por demostración (PpD) es una técnica, donde, al contrario de la programación detallada que convencionalmente se hace al robot, este aprende movimientos parciales o incluso una tarea completa, con solo ver una o varias demostraciones de los movimientos o la tarea por parte de un humano u otro robot.

El manejo de herramientas por parte de un robot, se puede dividir en subtareas como reconocimiento de la herramienta, agarre y posicionado de la herramienta, o reproducción de las acciones de manipulación de la herramienta para realizar la meta de la tarea. Estas subtareas, pueden ser aprendidas por demostración.

Se presenta una técnica que permite llevar una herramienta hasta la cabeza de un tornillo siguiendo una trayectoria similar a las demostradas por un ser humano. Se utiliza un robot humanoide NAO (Fig. 1) con capacidad de visión. Empleando técnicas de visión por computador y modelos probabilísticos, se logra flexibilidad en la reproducción de la tarea, ya que esta permite alinear la llave con la cabeza del tornillo, aún si se presentan variaciones en la ubicación y orientación de la cabeza del tornillo. Esta información visual es una de las entradas a un modelo probabilístico de las trayectorias, con el que se logra generar una nueva trayectoria.

El robot NAO [2], es un robot humanoide programable de 58 centímetros de alto, desarrollado por la compañía francesa Aldebaran. Entre sus principales características se encuentran: un cuerpo con 25 grados de libertad y una red de sensores, incluyendo dos cámaras RGB. Aunque se utilizó un robot humanoide, la técnica puede ser adaptada a robots industriales y la contribución consiste en la aplicación de la técnica de modelos probabilísticos en el manejo de herramientas.

En la sección 2 se presentan los trabajos relacionados. En la sección 3 se muestran los detalles de la implementación de la técnica. Primero, se muestra el procesamiento de las imágenes para obtener la posición y orientación de la tuerca, luego la técnica probabilística usada para lograr la flexibilidad en la reproducción de la tarea. En la sección 4, se presenta el experimento, gráficas de las trayectorias de demostración capturadas y las nuevas trayectorias logradas, además se discuten los resultados. Finalmente se concluye el documento.

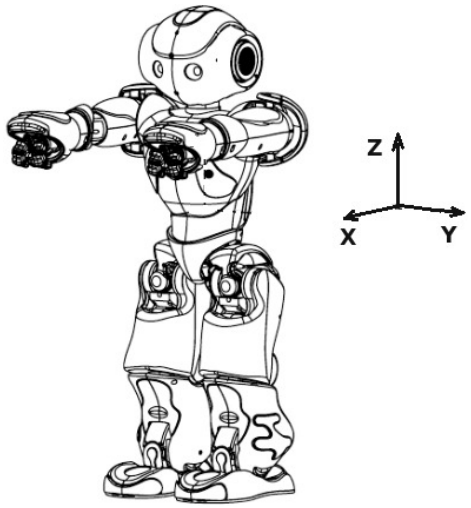


Fig. 1. Robot NAO. Fuente: [1]

2. TRABAJOS RELACIONADOS

Los primeros trabajos en programación por demostración, también llamados aprendizaje por imitación [3]-[5] surgen como una posible solución, o ruta alterna, que evita la tarea de programar el robot para tareas complejas. Un artículo de revisión desde un punto de vista no matemático, puede encontrarse en [6] y un capítulo sobre programación de robots por demostración es el presentado en [4].

El problema de enseñar a un robot el manejo de herramientas usando demostraciones, ha sido tratado por algunos autores. En [7], [8] logran que el robot Robonaut aprenda el manejo de herramientas, entre ellas un atornillador eléctrico (similar a un taladro), a partir de una o varias teleoperaciones. Usando visión estéreo y el

operador Laplaciano de la Gaussiana, obtienen la pose del atornillador eléctrico y realizan el agarre de este en cuatro pasos, los cuales consisten en movimientos de la palma y los dedos. La parte de manejo de la herramienta, es primero realizada por tele-operación, pero el robot aprende del humano y luego el robot es capaz de volverla a ejecutar cuando se le ordena por parte de un humano. En cuanto al problema de la orientación de la cabeza de la tuerca, los autores solo proponen posibles técnicas de solución.

En [9] se presenta una técnica que combina programación por demostración y control servo visual. Usando imágenes 2D y visión estéreo, detectan la punta de una herramienta y a través de la demostración del uso de la herramienta, adquieren los movimientos de esta punta, luego agregando la herramienta como un enlace adicional del modelo cinemático del brazo del robot, y usando control servo visual, el robot reproduce los movimientos humanos. El control servo visual permite flexibilidad en el largo de la herramienta. Los autores solo presentan resultados del manejo de una herramienta y no abordan problemas como encajar una llave en la cabeza de un tornillo.

En [10] utilizan modelos ocultos de Markov y una técnica de modelos inversos (HAMMER) para reconocer el uso de herramientas, mas que reconocer la forma de la herramienta, reconocen los movimientos que se hacen con esta, aunque no presentan la reproducción por parte del robot.

Brown y Sammunt [11] presentan la simulación de un robot móvil con una pinza, el cual aprende a tomar con la pinza un gancho y con este sacar una caja de un pasillo cerrado. A través de demostraciones, el robot aprende la tarea para una llave, construyendo modelos gramaticales, luego a través de experimentos y aprendizaje activo puede generalizar el aprendizaje para otro tipo de llaves. Sin embargo, los autores no abordan el problema de cambios en la tarea.

3. METODOLOGÍA

La técnica implementada consta de dos partes: a) Obtención de la información visual (posición y orientación de la cabeza del tornillo), y b) generación de la trayectoria correspondiente a la información visual obtenida, la cual lleva el brazo del robot desde una posición de reposo hasta la cabeza del tornillo siguiendo una trayectoria que procura ser similar a la de las trayectorias demostradas. La trayectoria se genera usando una técnica que emplea modelos probabilísticos, llamada modelo de mezclas de Gaussianas parametrizado en la tarea (TPGMM) desarrollada por Calinon [12], basada en los modelos Gaussianos Parametrizados (PGMM) [13].

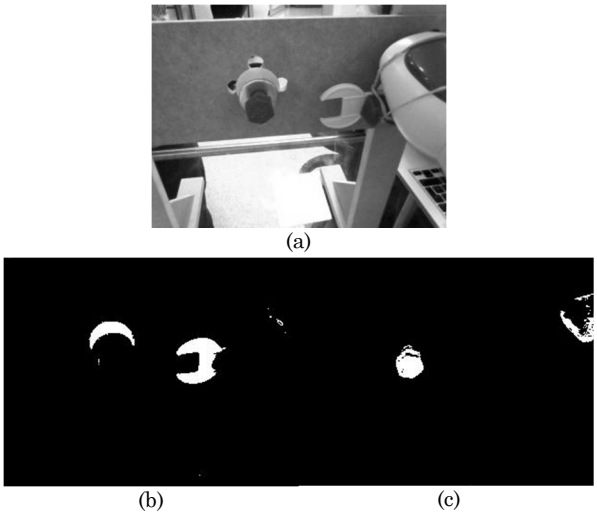
3.1 Obtención de la información visual de la cabeza del tornillo

A partir de la imagen de la cámara superior del robot NAO, se realizó el procesamiento de la imagen el cual consistió en: a) Segmentación en el espacio de color RGB, b) selección de puntos de interés de la cabeza del tornillo, y c) cálculo de la posición y orientación, con respecto a un marco de referencia. Información adicional se puede encontrar en [14].

3.1.1 Segmentación en color

Se contó con una iluminación controlada durante los experimentos, facilitando la segmentación en el espacio de color RGB. Luego de tomar varias imágenes de la herramienta y la tuerca que se quieren segmentar, se separa cada imagen en los canales rojo (R), verde (G) y azul (B) que lo constituyen. Mediante un software que permite leer el valor del píxel seleccionado, se obtuvieron los rangos de nivel de gris entre máximo y mínimo para cada objeto en cada canal. El resultado es una imagen binaria que contiene tanto los objetos segmentados como áreas de ruido producidas durante la segmentación (Fig. 2). De esta imagen se obtienen los contornos, en los

que pueden calcularse los momentos estadísticos [15]. Específicamente, el momento espacial m_{00} se refiere al área del contorno y se utiliza para realizar un filtrado del ruido dejando sólo el contorno de mayor área, es decir, el objeto que está siendo segmentado.

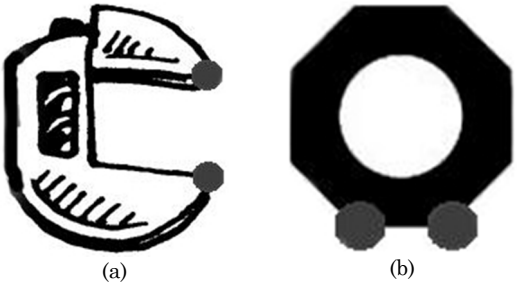


(a) (b) (c)

Fig. 2. Imágenes obtenidas luego de segmentar la imagen original (a), la llave (b), y la tuerca (c). Fuente: Autores

3.1.2 Selección de puntos de interés

Se definen los objetos a caracterizar como una llave y una tuerca (Fig. 3). Para cada punto en el contorno resultante, se calcula el ángulo que genera con respecto a los puntos anterior y posterior, a medida que se recorre el borde del objeto. Para esto, se calculan los vectores($\overrightarrow{v_1}$y$\overrightarrow{v_2}$) mostrados en la Fig. 4, que van desde el punto anterior (*j - 1*) y posterior (*j + 1*) hasta el punto que se está procesando (*j*). En (1), se indica cómo se realiza el cálculo del ángulo entre los dos vectores.



(a) (b)

Fig. 3. Ejemplo de los objetos a caracterizar: a) Llave y b) Tuerca. En gris los puntos de interés. Fuente: Autores

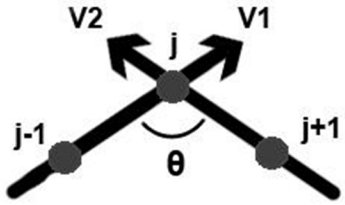
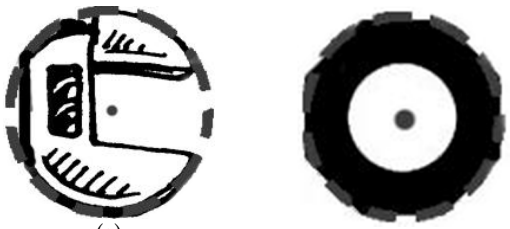


Fig. 4. Puntos utilizados para calcular el ángulo en un punto del contorno, con respecto a los puntos anterior y posterior. Fuente: Autores

$$\theta = arcos\left(\frac{\overrightarrow{v_1}.\overrightarrow{v_2}}{\|\overrightarrow{v_1}\|\|\overrightarrow{v_2}\|}\right) \quad (1)$$

En el caso de la llave, los ángulos deben permitir hallar los puntos correspondientes a las salientes de la herramienta (Fig. 3a), mientras que en la tuerca se desea encontrar dos aristas de la parte inferior (Fig. 3b). Se usó la librería para el procesamiento de imágenes OpenCV [16]. Entre las funciones de descripción de forma y análisis estructural, se encuentra la función *minEnclosingCircle*, la cual encuentra el círculo de área mínima que encierra un contorno (Fig. 5). Con la información del centro y el radio del círculo, se limitan el número de puntos encontrados con anterioridad, restringiendo áreas que no son de interés.



(a) (b)

Fig. 5. Ejemplo de usar la función de círculo de área mínima que encierra el objeto: a) Llave y b) Tuerca. Fuente: Autores

Debido a que el robot NAO sostiene la llave con la mano derecha, se generan restricciones dadas por la estructura del robot. Específicamente, la llave nunca estará apuntando hacia la derecha o hacia abajo. Como se observa en las Fig. 6a y Fig. 6b, puntos encontrados en la sección señalada no se tienen en cuenta (sección inferior derecha del contorno).

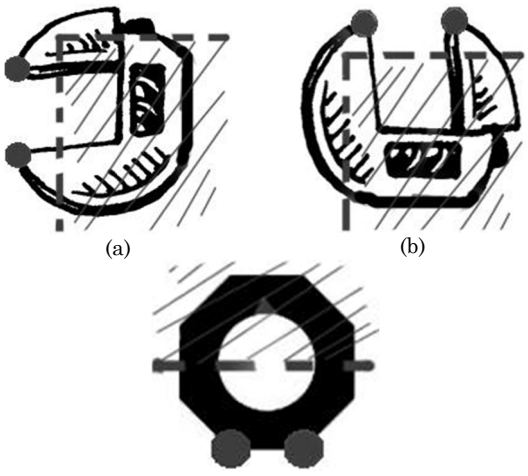


Fig. 6. Restricciones implementadas para la selección de los puntos de interés. Fuente: Autores

En el caso de la tuerca (Fig. 6c), solo puntos ubicados en la parte inferior del contorno son seleccionados. De las múltiples aristas encontradas en este sector, se seleccionan las dos ubicadas más abajo en la imagen.

3.1.3 Cálculo de la orientación de los objetos

Para determinar la orientación, primero se define un marco de referencia sobre cada objeto. Los ángulos quedan definidos de tal forma, que tanto la llave como la tuerca, se alineen cuando el ángulo es el mismo en ambos objetos. Es decir, la orientación de la tuerca debe igualarse en la llave para permitir que la herramienta pueda realizar una acción sobre el elemento.

Tanto el ángulo de la llave como de la tuerca se calculan utilizando (1), donde uno de los vectores se define como la línea que atraviesa los dos puntos de interés definidos precedentemente sobre el objeto. El segundo vector se define con el marco de referencia. En el caso de la llave, se utiliza el punto de interés que se encuentra más a la izquierda y a partir de este se genera un vector hacia arriba de la imagen (Fig. 7a); en el caso de la tuerca, se toma el punto de interés más bajo y a partir de este se genera un vector hacia la derecha (Fig. 7b).

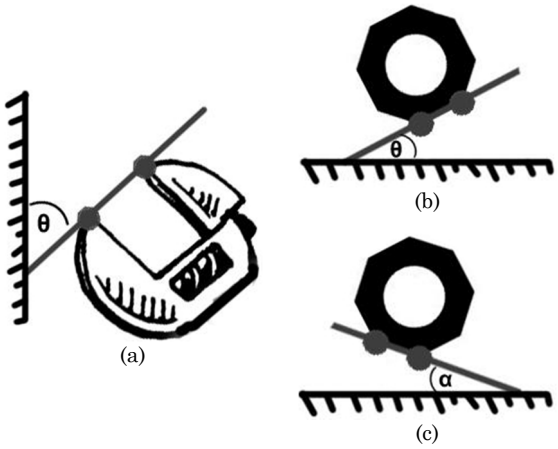


Fig. 7. Marco de referencia para definir la orientación de los objetos. Fuente: Autores

Para el cálculo de la orientación de la tuerca existe una consideración especial. Dado que los puntos de interés se definen como las aristas de la tuerca que se encuentran en una posición más baja, existe la posibilidad de que el ángulo que se calcula sea α, como lo describe la Fig. 7c. Para calcular el ángulo deseado θ (Fig. 7b), se realiza la corrección que se muestra en (2), donde 50º corresponde al ángulo interior de las aristas de la tuerca.

$$\theta = 50^o - \alpha \quad (2)$$

3.2 Generación de la trayectoria de la tarea

En la Fig. 8 se muestran las tres etapas del proceso de generación de la trayectoria. En una primera etapa se adquiere la información de las trayectorias de las demostraciones y parámetros de la tarea, luego mediante un algoritmo EM (Maximización de la Esperanza) modificado se obtienen los parámetros del modelo, usando una técnica que combina sistemas dinámicos y regresión de mezclas de Gaussianas (DS-GMR), presentada en [12]. En el último bloque se calcula la nueva trayectoria, a partir del modelo estimado y los nuevos parámetros de la tarea; esta trayectoria es similar a las trayectorias demostradas. A continuación, se describe el procedimiento.

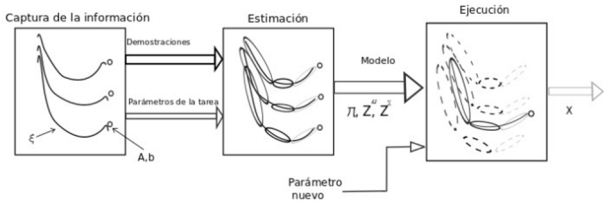


Fig. 8. Diagrama de bloques de la técnica. Fuente: Autores

3.2.1 Bloque de captura de la información

A partir de M trayectorias demostradas $\mathcal{J}i$ con $i=1...M$, cada una con T_m puntos, se forma un conjunto de datos $\boldsymbol{\xi}$ de N puntos. Con lo que $N = \sum_{m=1}^{M} T_m$. Cada punto del conjunto, consta de un dato de tiempo y la respectiva posición cartesiana en ese punto $\xi_n = [t_n, \boldsymbol{p}_n]$, con n un punto específico.

Para cada punto es necesario conocer N_p parámetros de la tarea (marcos de referencia), que consisten en posiciones $\boldsymbol{b}_{n,j}$ y matrices de orientación $\boldsymbol{A}_{n,j}$. Los índices n y j representan la muestra de tiempo n y el marco j-ésimo de referencia.

Para cada marco de referencia, existen unos parámetros del modelo, los cuales son modelos locales con respecto a cada marco. El producto de las gaussianas respectivas entre los marcos, es el que finalmente modela las trayectorias.

3.2.2 Bloque de estimación

Este bloque a su vez consta de una inicialización y de una estimación de los parámetros del modelo, usando un algoritmo de inicialización, se calculan los parámetros del modelo $\{\boldsymbol{\pi}, \boldsymbol{Z}^{\mu}, \boldsymbol{Z}^{\Sigma}\}$ preliminares, usando (3):

$$\{\boldsymbol{\pi}_0, \boldsymbol{Z}_0^{\mu}, \boldsymbol{Z}_0^{\Sigma}\} = inicializacion(\boldsymbol{\xi}, \boldsymbol{b}, \boldsymbol{A}) \quad (3)$$

Donde $\boldsymbol{\pi}$ son los coeficientes de mezcla, $\boldsymbol{Z}^{\mu}$ la matriz de promedios y las $\boldsymbol{Z}^{\Sigma}$ son las matrices de covarianza, y el algoritmo de inicialización puede ser basado en divisiones en el tiempo, o por *k-means* (modelo por grupos de datos). A partir del resultado de la inicialización, se refina el modelo, usando una modificación propuesta en [12] del procedimiento EM, usando (4):

$$\{\boldsymbol{\pi}, \boldsymbol{Z}^{\mu}, \boldsymbol{Z}^{\Sigma}\} = EM\_modificado\left(\boldsymbol{\xi}, \boldsymbol{b}, \boldsymbol{A}, \boldsymbol{\pi}_0, \boldsymbol{Z}_0^{\mu}, \boldsymbol{Z}_0^{\Sigma}\right) \quad (4)$$

La modificación con respecto al EM tradicional consiste en que el algoritmo tiene en cuenta los parámetros de la tarea, 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), se usa (5) con los parámetros temporales $\boldsymbol{\mu}_{n,i}$, $\sum_{n,i}$ los cuales se describen posteriormente, para calcular la probabilidad $h_{n,i}$. En el paso (M), los parámetros del modelo son estimados a partir de esta probabilidad, usando (6) a (9):

Paso E:

$$h_{n,i} = \frac{\pi_i \mathcal{N}(\boldsymbol{\xi}_n | \boldsymbol{\mu}_{n,i}, \boldsymbol{\Sigma}_{n,i})}{\sum_{k=1}^{N_k} \pi_k \mathcal{N}(\boldsymbol{\xi}_n | \boldsymbol{\mu}_{n,k}, \boldsymbol{\Sigma}_{n,k})}, \quad (5)$$

donde $\mathcal{N}(\,)$ es la función Gaussiana.

Paso M:

$$\pi_i = \frac{\sum_{n=1}^{N} h_{n,i}}{N}, \quad (6)$$

$$\boldsymbol{Z}_{i,j}^{\mu} = \frac{\sum_{n=1}^{\mathrm{N}} h_{n,i} \boldsymbol{A}_{n,j}^{-1} [\boldsymbol{\xi}_n - \boldsymbol{b}_{n,j}]}{\sum_{n=1}^{\mathrm{N}} h_{n,i}}, \quad (7)$$

$$\boldsymbol{Z}_{i,j}^{\Sigma} = \frac{\sum_{n=1}^{N} h_{n,i} \boldsymbol{A}_{n,j}^{-1} [\boldsymbol{\xi}_n - \tilde{\boldsymbol{\mu}}_{n,i,j}][\boldsymbol{\xi}_n - \tilde{\boldsymbol{\mu}}_{n,i,j}] \boldsymbol{A}_{n,j}^{-T}}{\sum_{n=1}^{N} h_{n,i}}, \quad (8)$$

$$\tilde{\boldsymbol{\mu}}_{n,i,j} = \boldsymbol{A}_{n,j} \boldsymbol{Z}_{i,j}^{\mu} + \boldsymbol{b}_{n,j} \quad (9)$$

Los parámetros temporales $\boldsymbol{\mu}_{n,i}$, $\sum_{n,i}$ corresponden a la matriz de promedios y de covarianza resultante, la cual es el producto de las Gaussianas calculadas con los j-ésimos parámetros del modelo y los parámetros de la tarea, usando (10):

$$\mathcal{N}(\boldsymbol{\mu}_{n,i}, \boldsymbol{\Sigma}_{n,i}) = \prod_{j=1}^{N_p} \mathcal{N}(\boldsymbol{A}_{n,j} \boldsymbol{Z}_{i,j}^{\mu} + \boldsymbol{b}_{n,j}, \boldsymbol{A}_{n,j} \boldsymbol{Z}_{i,j}^{\Sigma} \boldsymbol{A}_{n,j}^{T}) \quad (10)$$

3.2.3 Bloque de ejecución

En este bloque, a partir del modelo y los parámetros, se genera la nueva trayectoria, usando (11):

$$\boldsymbol{x} = reproduccion(\boldsymbol{t}, \{\boldsymbol{\pi}, \boldsymbol{Z}^{\mu}, \boldsymbol{Z}^{\Sigma}\}, \{\boldsymbol{A}, \boldsymbol{b}\}), \quad (11)$$

Donde $\boldsymbol{t}$, es un vector de valores en el tiempo, $\boldsymbol{x}$ son los valores cartesianos de la trayectoria, y $\{\boldsymbol{A}, \boldsymbol{b}\}$ son los parámetros de la tarea que se refieren o guardan relación con la nueva trayectoria que se desea generar.

3.2.4 Composición de parámetros de la tarea

Los parámetros de la tarea guardan la información sobre variaciones de posición y dirección de los puntos de referencia. Están compuestos por la matriz de transformación $\boldsymbol{A}$ (dirección) y el vector de desplazamiento $\boldsymbol{b}$ (posición). Se deben calcular, tanto en el proceso de estimación del modelo para cada una de las trayectorias, como cuando se pretende calcular una nueva trayectoria. El cálculo de $\boldsymbol{A}$ y $\boldsymbol{b}$, se realiza a partir de dos puntos en el espacio cartesiano $\boldsymbol{p}_1$ y $\boldsymbol{p}_2$. La matriz de transformación $\boldsymbol{A}$, se calcula usando (12) a partir de tres vectores $\boldsymbol{u}_1$, $\boldsymbol{u}_2$ y $\boldsymbol{u}_3$. El vector $\boldsymbol{u}_2$ es la diferencia $\boldsymbol{p}_1$ - $\boldsymbol{p}_2$ normalizada y los vectores $\boldsymbol{u}_1$, $\boldsymbol{u}_3$ son perpendiculares a $\boldsymbol{u}_2$, 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 $\boldsymbol{p}_1$ usando (13):

$$\boldsymbol{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & u_{1x} & u_{2y} & u_{3y} \\ 0 & u_{1y} & u_{2y} & u_{3y} \\ 0 & u_{1z} & u_{2y} & u_{3y} \end{bmatrix} \quad (12)$$

$$\boldsymbol{b} = [0 \quad p_{1x} \quad p_{1y} \quad p_{1z}] \quad (13)$$

4. RESULTADOS Y DISCUSIÓN

En la Fig. 9 se muestra el robot NAO y el tablero que se utilizó para los experimentos. Este último tiene un bloque de espuma con diversos agujeros que permiten colocar el tornillo en varios puntos. Aunque el robot tiene un procesador, sus capacidades son limitadas, por lo que se optó por usar un ordenador remoto, para el procesamiento del vídeo producto de las cámaras del robot.

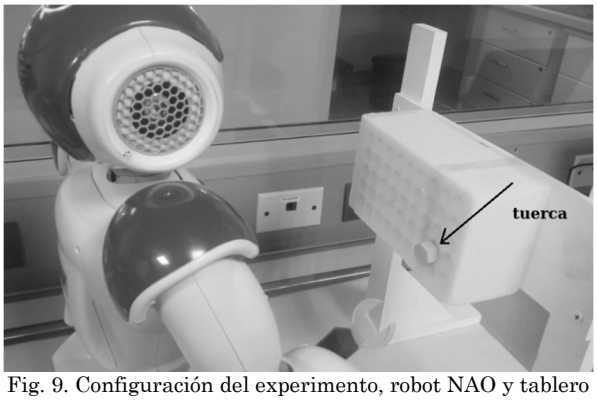


Fig. 9. Configuración del experimento, robot NAO y tablero de pruebas. Fuente: Autores

4.1 Obtención de las demostraciones y parámetros de la tarea

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

4.2 Resultados estimación del modelo

Para la estimación del TPGMM, el número de Gaussianas se tomó de manera heurística en seis, y se usaron dos parámetros: a) Posición inicial de la trayectoria demostrada y b) posición final. El primer parámetro, o dato inicial, se calcula a partir de los puntos $\boldsymbol{p}_1$ y $\boldsymbol{p}_2$. El punto $\boldsymbol{p}_{11}$ (Fig. 10a) es el dato inicial de cada trayectoria, y

$\boldsymbol{p}_{12}$ se toma manualmente unas posiciones después de iniciar la trayectoria. Para el segundo parámetro o dato final, el punto $\boldsymbol{p}_{21}$, se tomó como el final de cada trayectoria y el punto $\boldsymbol{p}_{22}$, se tomó manualmente unas posiciones antes de terminar la trayectoria. En las Fig. 10b y 11, 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 Fig. 1.

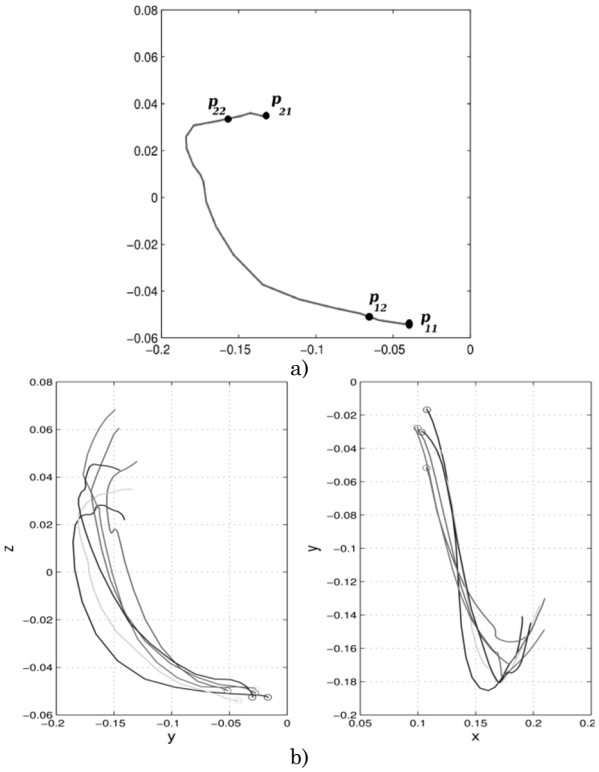


Fig. 10. Trayectorias de las demostraciones capturadas con el mismo robot. Los círculos indican el inicio de la trayectoria. a) Obtención de los puntos p_1 y p_2. b) Trayectoria demostradas planos yz y xy. Fuente: Autores

4.3 Generación de la nueva trayectoria

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), $\boldsymbol{p}_{11}$ se definió como el promedio de los datos de inicio de todas las trayectorias demostradas, y $\boldsymbol{p}_{12}$ se definió de manera similar al anterior. Para el segundo parámetro (punto final), $\boldsymbol{p}_{21}$ se obtuvo a partir de la posición del centro de la cabeza del tornillo calculada por visión, más un dato de traslación, que lo lleva al marco de referencia de la mano del robot. El punto $\boldsymbol{p}_{22}$, se obtuvo a partir del cálculo del punto $\boldsymbol{p}'_{22}$ (Fig. 12), adicionándole la misma translación comentada para $\boldsymbol{p}_{21}$. El punto $\boldsymbol{p}'_{22}$, se obtiene como el punto a una distancia L, sobre la línea que parte del centro del tornillo calculado por visión, y que tiene un ángulo de orientación θ de la cabeza del tornillo (ver sección 3).

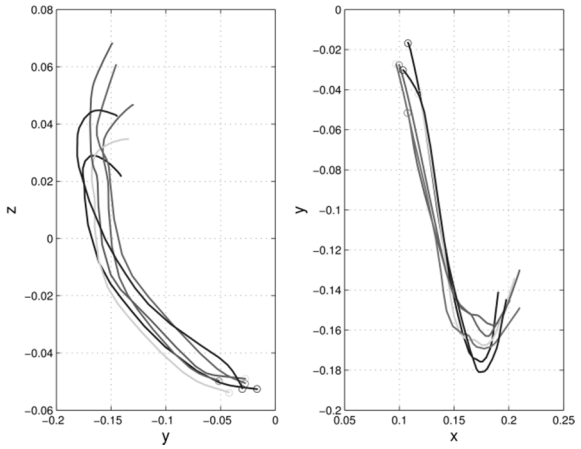


Fig. 11. Trayectorias de las reproducciones usando los mismos parámetros de la tarea de las trayectorias demostradas. Los círculos indican el inicio de la trayectoria. Fuente: Autores

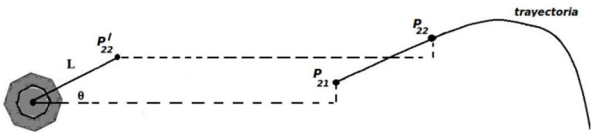


Fig. 12. Cálculo de los puntos$\boldsymbol{p}'_{22}$ y $\boldsymbol{p}_{22}$. Fuente: Autores

En la Fig. 13, se muestran dos ejemplos de trayectorias nuevas generadas (línea gruesa). En la Fig. 14, se muestra una comparación del error RMS calculado entre una trayectoria demostrada $\boldsymbol{\mathcal{T}}$ y la trayectoria reproducida $\boldsymbol{x}$ respectiva, usando (14):

$$e_{RMS} = \frac{1}{T_m}\sum_{t=1}^{T_m} \|\boldsymbol{\mathcal{T}}_t - \boldsymbol{x}_t\| \tag{14}$$

En la Fig. 15, se muestra una secuencia del robot ejecutando una de las trayectorias.

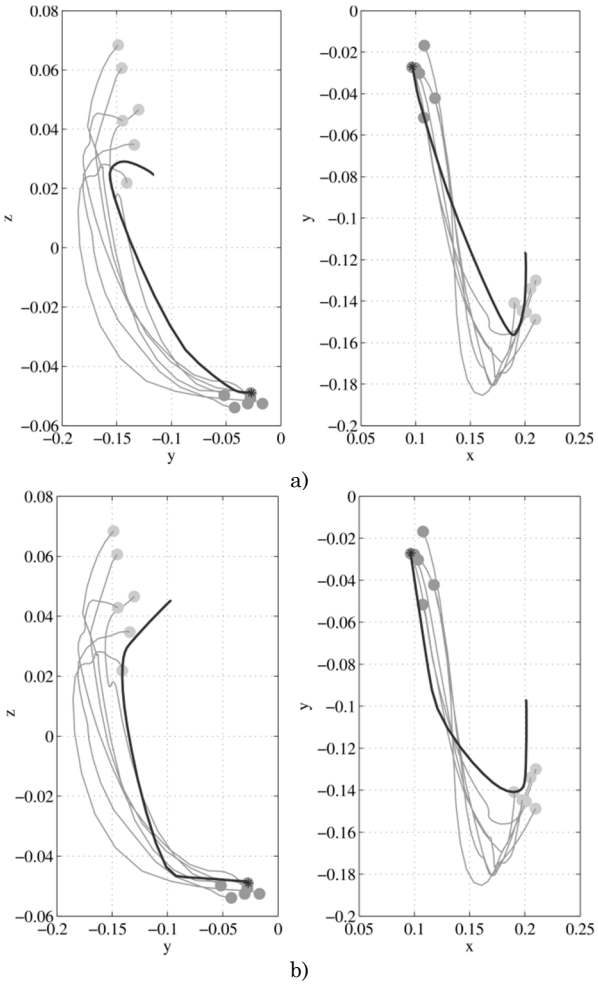


Fig. 13. Ejemplos de generación de nuevas trayectorias. En gris las demostraciones, en negro, dos ejemplos de trayectorias reproducidas para nuevos parámetros.
Fuente: Autores

4.4 Discusión de resultados y problemas detectados

El modelado con TPGMM distorsiona un poco las trayectorias, sobretodo en xy, lo anterior se puede mejorar agregando más Gaussianas. Esto no se realizó para evitar el aumento de tiempo de estimación del modelo, además como puede verse en la Fig. 14a, se puede ver que el aumento en número de Gaussianas, tampoco reduce mucho el error RMS.

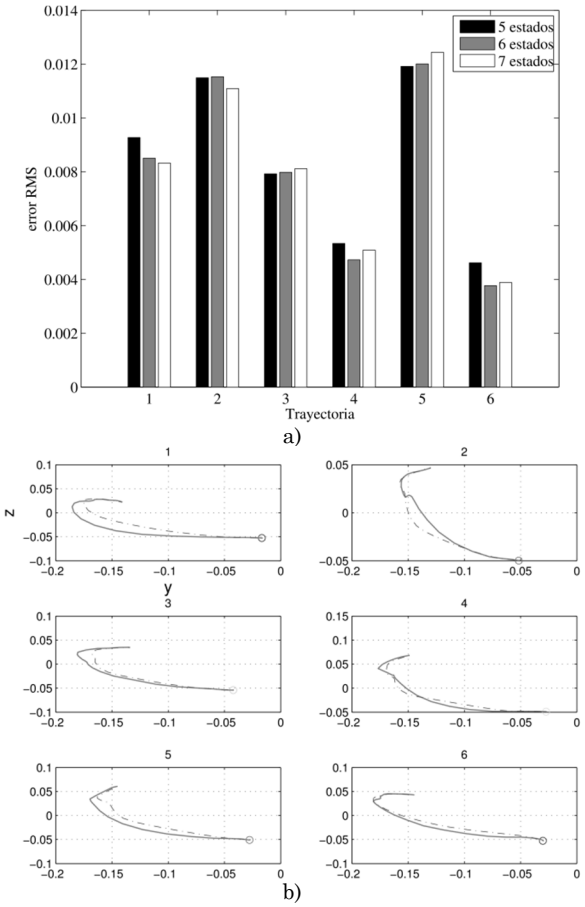


Fig. 14. a) Comparación del error RMS al aumentar el número de estados o Gaussianas. b) Trayectorias demostradas (línea continua), trayectorias reproducidas usando 6 estados (línea punteada). Fuente: Autores

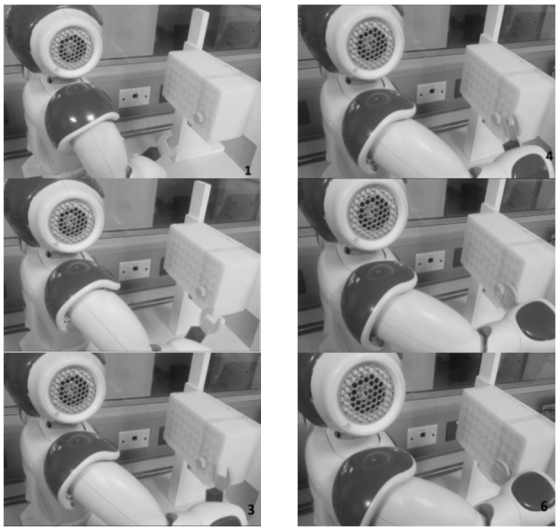


Fig. 15. Secuencia de la tarea de llevar la llave, a la cabeza del tornillo. Fuente: Autores

De la Fig.14b se puede ver que las trayectorias reproducidas terminan con un ángulo muy similar, lo que permite que la mano del robot ubique la llave aproximadamente con la orientación adecuada. Pero, en algunas pruebas, se presentó un error en la aproximación de la mano-llave a la cabeza del tornillo, este al parecer se debe a errores en el sistema de visión y juego en la unión del torso con las piernas. El primero es debido a la resolución de la cámara del robot y a errores de calibración. El segundo, es que a pesar de que se colocaban rígidas las articulaciones del torso y las piernas, se presentaban desplazamientos en la posición final de la mano.

Se intentó una ubicación diferente del tornillo, en un plano paralelo a las coordenadas xy del robot (como una mesa), pero los 4 grados de libertad del brazo del robot NAO, no permiten físicamente guiar el brazo para obtener las trayectorias de demostración requeridas.

5. CONCLUSIONES

Una aplicación de programación por demostración en robótica se presenta, la técnica combina modelos probabilísticos y visión para realizar una tarea que permite a un robot humanoide llevar una herramienta (llave) hasta la cabeza de una tuerca. Mediante algunos resultados, se presenta su funcionamiento y los principales problemas presentados.

Pese a algunas limitaciones de la plataforma robótica NAO, se lograron resultados aceptables. Los problemas presentados, descritos en la sección anterior provocan que en algunos casos, la llave no se inserte en la cabeza de la tuerca.

En lo referente a trabajo futuro, se podría disminuir el error obtenido en el cálculo de la orientación de los objetos, aumentando la resolución de las imágenes obtenidas del robot de 320x240 a 640x480 píxeles, a expensas de aumentar el tiempo de cómputo. También, en trabajos posteriores, se propone utilizar una técnica que además de programación por demostración, agregue funciones de visión y medida de fuerzas para realizar la tarea de apretar una tuerca. El robot a partir de la demostración aprende el movimiento de apretar, con la visión se logra flexibilidad de la tarea demostrada ya que permite alinear la llave con la cabeza del tornillo, y por último, se harán mediciones de corriente para identificar el punto donde la tuerca ha sido completamente enroscada.

6. AGRADECIMIENTOS

El proyecto presentado está soportado por la Universidad Nacional de Colombia y la Universidad de La Sabana, bajo el código ING-134-2013: "Continuación de tareas por un robot usando aprendizaje por imitación".

7. REFERENCIAS

[1] N. Kofinas, "Forward and Inverse Kinematics for the NAO Humanoid Robot," Chania, 2012.

[2] D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier, "Mechatronic design of NAO humanoid," in *2009 IEEE International Conference on Robotics and Automation*, 2009, pp. 769–774.

[3] S. Schaal, "Is imitation learning the route to humanoid robots?," *Trends Cogn. Sci.*, vol. 3, no. 6, pp. 233–242, Jun. 1999.

[4] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot Programming by Demonstration," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer, 2008, pp. 1371–1394.

[5] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Rob. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, May 2009.

[6] C. Breazeal and B. Scassellati, "Robots that imitate humans," *Trends Cogn. Sci.*, vol. 6, no. 11, pp. 481–487, Nov. 2002.

[7] W. Bluethmann, R. Ambrose, M. Diftler, S. Askew, E. Huber, M. Goza, F. Rehnmark, C. Lovchik, and D. Magruder, "Robonaut: a robot designed to work with humans in space.," *Auton. Robots*, vol. 14, no. 2–3, pp. 179–97, 2003.

[8] W. Bluethmann, R. Ambrose, M. Diftler, E. Huber, A. Fagg, M. Rosenstein, R. Platt, R. Grupen, C. Breazeal, A. Brooks, A. Lockerd, R. A. Peters, O. C. Jenkins, M. Mataric, and M. Bugajska, "Building an autonomous humanoid tool user," in *4th IEEE/RAS International*

Conference on Humanoid Robots, 2004., 2004, vol. 1, pp. 402–421.

[9] A. Edsinger and C. C. Kemp, "Toward robot learning of tool manipulation from human demonstration," 2007.

[10] Y. Wu and Y. Demiris, "Learning dynamical representations of tools for tool-use recognition," in *2011 IEEE International Conference on Robotics and Biomimetics*, 2011, pp. 2664–2669.

[11] S. Brown and C. Sammut., "Tool use learning in robots," in *2011 AAAI Fall Symposium Advances in Cognitive Systems*, 2011, pp. 58–65.

[12] S. Calinon, Z. Li, T. Alizadeh, N. G. Tsagarakis, and D. G. Caldwell, "Statistical dynamical systems for skills acquisition in humanoids," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, 2012, pp. 323–329.

[13] A. D. Wilson and A. F. Bobick, "Parametric hidden Markov models for gesture recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 9, pp. 884–900, 1999.

[14] C. Peña, J. Hoyos, F. Prieto, J. Ayala, and C. Garzon-Castro, "Screw and wrench orientation estimation using the artificial vision in the NAO humanoid robot," in *2013 II International Congress of Engineering Mechatronics and Automation (CIIMA)*, 2013, pp. 1–6.

[15] Y. C. Chim, A. A. Kassim, and Y. Ibrahim, "Character recognition using statistical moments," *Image Vis. Comput.*, vol. 17, no. 3–4, pp. 299–307, Mar. 1999.

[16] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. O'reilly, 2008, p. 543.