Proyecto Final. Conducción Autónoma

Linette Zamudio Guzmán
151235
linette.zamudio@itam.mx

Alberto Salinas Chávez
149038
asalin10@itam.mx

Octavio Oropeza Montalvo
147727
ooropeza@itam.mx

Guillermo Flores Cuevas
150951
gflore14@itam.mx

***Abstract*—El siguiente documento contiene información sobre el desarrollo de un programa que permite la estimación de estados tanto propio como de un óbstaculo móvil donde su objetivo es implementar componentes útiles para la conducción autónoma del robot AutoNOMOs apoyados en ROS (Robot Operating System) y Gazebo.**

I. Introducción

El objetivo de este proyecto es aplicar la teoría desarollada a lo largo del curso sobre estimación de estados y control de trayectoria para poder definir el estado propio y el de un obstáculo móvil, de tal forma que se logra simular el comportamiento de un carro autónomo a la hora de seguir a dicho obstáculo móvil.

Se divide en tres secciones principales.
Estimación de estado propio - en el cual se estima en qué carril está el robot utilizando un filtro de histogramas.
Estimación del estado de un obstáculo móvil - Con base en una nube de puntos entregados por un LiDAR se logra estimar la velocidad a la que se mueve un objeto externo con respecto al robot
Movimiento para seguir al obstáculo móvil - Control que permita el movimiento con la especificación del modelo de Ackermann, el cual ayudará a la creación de la trayectoria óptima. Dicha trayectoria será modificada a través de la velocidad y dirección de movimiento del robot haciendo uso del filtro de Kalman.

Este problema puede ser común dentro de la robótica autónoma, sin embargo representa un gran reto de implementación. El modificar la trayectoria de un robot para que logre seguir a un objeto es considerado el punto de partida para que, a partir de esto, se pueda crear un robot autónomo con mayores y mejores características.

Desglosando las aportaciones de cada integrante al proyecto se tiene lo siguiente:

1) **Alberto Salinas** Pseudocódigo del filtro de Kalman y el control para que el auto se mueva a un punto deseado.
2) **Guillermo Flores** Implementación del código en C++ e interpretación de los datos arrojados por los distintos nodos, principalmente los más importantes para efecto de llegar a una solución como fueron las lecturas de las rectas de la rosbag y del sensor LiDAR del Autonomos_mini_1.
3) **Linette Zamudio** Experimentación de cada una de las partes del proyecto, incluyendo la creación del código que permita la lectura de la pose del obstáculo, así como el nodo states_node que arroja las probabilidades de la primera parte del proyecto.
4) **Octavio Oropeza** Pseudocódigo del filtro de histogramas para la predicción de los estados siguientes del automóvil.

   Entre todos se realizó el escrito.

II. Marco Teórico

Parte 1 Correspondientes a los temas del primer examen parcial

1) Represente la pose del obstáculo con respecto al robot en función de las lecturas del LiDAR

   De acuerdo con las especificaciones del sensor LiDAR se obtienen lecturas de aproximadamente los 360 grados alrededor del robot que representan las distancias a ciertos objetos, sin embargo, por como está acomodado, se registran distancias a los postes del robot en cuestión. Esto representa un problema a la hora de determinar la posición pues son valores que no deben tomarse en cuenta al no corresponder a las distancias del obstáculo. Las lecturas del LiDAR obtenidas por el tipo de mensaje std_msgs/Laser_scan que contiene dos arreglos importantes de valores; rangos e intensidades. Para efectos de nuestro objetivo el último arreglo no se utiliza para representar la pose, por lo que omitió el procesamiento del mismo. Ahora bien, el arreglo de rangos es el que, como tal, contiene las distancias medidas a los distintos ángulos que el LiDAR nos da. Obteniendo todas las medidas en un arreglo de 360 posiciones (una por cada incremento en ángulo), se declara una función que genera el ángulo correspondiente a la distancia medida, por lo que entre ángulo y ángulo, la distancia entre ambos es el incremento que dan los rayos desplegados por el LiDAR. A partir de esto, se crea otro arreglo para definir todos los ángulos posibles que hiciera posible establecer la relación entre la distancia y su correspondiente ángulo. Como ya se mencionó, los postes del robot en cuestión dieron valores que no debían tomarse en cuenta a la hora de realizar la representación de la pose, por lo que, para solucionar esto, fue necesario establecer una condición que despreciara dichas lecturas. Si la distancia resulta mayor a 0.13 representan un obstáculo externo, se toman en cuenta y se transforman esas coordenadas polares a cartesianas para obtener la pose

del obstáculo respecto al robot mini_1.

2) Proponga y justifique una estrategia de control para que el robot iguale la velocidad del obstáculo móvil una vez que cuenta con la estimación de estado del mismo.
De acuerdo a las lecturas del LiDAR, resulta más sencilla la determinación del control de nuestro robot a través de *Moving to Point* porque se cuenta con las distancias referentes al obstáculo que permiten establecer el punto deseado en cada momento. La situación dinámica tanto del robot como del obstáculo móvil, implica que los otros controles sean de mayor complejidad a la hora de la implementación, por lo que el control elegido realiza de forma correcta la variación de los parámetros, es decir, el control. Este varía tanto la velocidad como el ángulo hacia la meta de la siguiente forma:

$$\theta^* = \tan^{-1}\frac{y^* - y}{x^* - x}$$

$$v^* = K_v\sqrt{(x^* - x)^2 + (y^* - y)^2}$$

De donde se puede observar la dependencia de la distancia que existe entre la posición actual y la deseada. Sin embargo, como se toma el sistema de coordenadas fijo al robot en cuestión, el par correspondiente a (x,y) de las fórmulas siempre son cero, facilitando el cálculo.

Parte 2 Correspondientes a los temas del segundo examen parcial

1) Investigue un algoritmo (en pseudo-código) que permita identificar rectas a partir de las lecturas del LiDAR. Descríbalo con detalle y explique como funciona.
La transformada de Hough es comúnmente utilizada para la detección de figuras en imágenes digitales dentro de las cuales podemos encontrar rectas o circunferencias. En el caso particular, el interés es por la detección de líneas rectas utlizando la transformación lineal.
Una recta puede representarse mediante pares (x,y) dentro de la imágen, en este caso, dichos pares son dados por el LiDAR, de tal forma que se tiene

$$y = m * x + n$$

El objetivo de la transformada de Hough es considerar dichas rectas en términos de los parámetros (m,n) , sin embargo, se indefinen en rectas verticales por lo que es mejor utilizar la representación en coordenadas polares mediante pares (ρ, θ) que correspondan a una curva sinusoidal en el espacio, donde:
ρ - Es la distancia entre el origen y el punto (x,y)
θ - Ángulo del vector que va desde el origen a la recta perpendicular de la original.
Quedando la ecuación de la recta de la siguiente manera:

$$y = \left(-\frac{\cos\theta}{\sin\theta}\right) * x + \left(\frac{\rho}{\sin\theta}\right)$$

La cual se puede reescribir

$$\rho = x * cos\theta + y * sin\theta$$

Por lo tanto, si las curvas correspondientes a dos puntos se intersecan, el punto de intersección que se forma en este nuevo espacio corresponde a una recta que pasa por esos dos puntos en la imagen original. El pseudo-código que representa lo anterior es el siguiente:

- Cargar la imagen
- Detectar los bordes en la imagen
- Por cada punto en la imagen:
  - Si el punto (x,y) esta en un borde:
    - Por todos los posibles angulos θ:
      - Calcular ρ para el punto (x,y) con un angulo θ
      - Incrementar la posicion (ρ,θ) en el acumulador
- Buscar las posiciones con los mayores valores en el acumulador
- Devolver las rectas cuyos valores son los mayores en el acumulador

2) Describa una secuencia de operaciones de procesamiento de imágenes que permitan identificar los puntos que corresponden a los carriles de la carretera y su mapeo a coordenadas en el plano.
Se consideraron dos secuencias que permitirían la resolución de la problemática presente. Por un lado se realiza la conversión a escala de grises de la imagen, de tal forma que sea posible identificar y relacionar el color blanco con las líneas de la carretera que se observa en la RosBag. Para lograr esto se debe aumentar el contraste de dicha conversión, y determinar los parámetros que serán considerados como el rango de color blanco, de tal forma que a la hora de codificar la detección de carriles sea más fácil la comparación con el resto de los colores en la imagen para determinar la ubicación de las líneas. Por otro lado, se aplica el proceso de segmentación de imágenes a través de Clusters, que a su vez, se divide en 3 problemas:

a) *Clasificación:* Proceso de decisión aplicado a cada pixel que asigna cada uno de estos a una clasificación específica, es decir una clase C. La clasificación siempre es específica de la aplicación, donde el objeto corresponde a pixeles blancos.
b) *Representación:*Los pixeles adyacentes de la misma clase están conectados entre sí para formar conjuntos espaciales.
c) *Descripción:* Los conjuntos se describen en términos de valores escalares o vectores. De lo anterior se tiene que la función *color_means* convierte primero cada pixel de color a su coordenada

xy_cromatidad. Cada pixel de color se asigna a un punto en el plano de cromaticidad xy.Luego, el algoritmo *k_means* se usa para encontrar grupos de puntos en el plano donde cada Cluster corresponde a un grupo de pixeles con un color distinguible, en este caso, el blanco.

Parte 3 Correspondientes a los temas de estimación de estado y aprendizaje

1) Explique el Filtro de Kalman y cómo lo utilizó en la estimación de estado del obstáculo móvil. Es un algoritmo recursivo en tiempo real que utiliza las mediciones del estado actual como entrada para predecir el estado siguiente de dicho sistema, es decir, genera un vector de estado estimado mediante la parametrización de momentos. Sin embargo, cada iteración conlleva una adición de ruido gaussiano que a su vez provoca una cierta incertidumbre a cada paso, dicha incertidumbre se representa por la adición de ruido gaussiano, el cual tiene las mismas dimensiones del vector de estados. El algoritmo del filtro es un proceso de dos pasos: El primer paso consigue adelantar acontecimientos, calculando de forma anticipada el estado xt mediante el estado anterior. El segundo paso utiliza las mediciones de ruido para actualizar dicha predicción, esta actualización busca corregir los posibles errores de forma que se obtenga una estimación lo más exacta posible, para ello, se cuenta con la ayuda de sensores que transmiten información necesaria del exterior, lo que permite comparar la predicción con lo percibido por los sensores. En el caso particular, se desea utilizar el filtro para lograr el seguimiento de objetos, de tal forma que permitan cumplir con el objetivo.
   El algoritmo del filtro que representa lo dicho anteriormente, es el siguiente:

Algorithm Kalman_filter$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$:
$\bar{\mu}_t = A_t\, \mu_{t-1} + B_t\, u_t$
$\bar{\Sigma}_t = A_t\, \Sigma_{t-1}\, A_t^T + R_t$
$K_t = \bar{\Sigma}_t\, C_t^T\, (C_t\, \bar{\Sigma}_t\, C_t^T + Q_t)^{-1}$
$\mu_t = \bar{\mu}_t + K_t(z_t - C_t\, \bar{\mu}_t)$
$\Sigma_t = (I - K_t\, C_t)\, \bar{\Sigma}_t$
return μ_t, Σ_t

Fig. 1. Algoritmo de Kalman

2) Explique el Filtro de Histogramas y cómo lo utilizó en la estimación de estado del robot con respecto a su carril. La función del filtro de histogramas es predecir un estado siguiente utilizando las probabilidades de encontarse en un estado en el tiempo 0 y las probabilidades de cambiar a otro estado en el tiempo 1 según las caracteristicas del estado en el tiempo 0, como lo fueron la presencia de rectas y el ángulo de giro "steering". Nuestra solución fue darle probabilidades a nuestros siete estados en función de las líneas que nos regresaba la Rosbag. Una vez que todos nuestros estados estaban definidos por las líneas, se creó otro arreglo de probabilidades en función del estado y de el steering leído, también proporcionado por la rosbag, en ese momento. Al convolucionar ambos arreglos se obtiene un nuevo arreglo de probabilidades que indica el estado más probable del vehículo en el tiempo 1, dichas probabilidades obtenidas del estado siguiente sirven para que el autonomous tome una mejor decisión basándose en ellas, por ejemplo, en caso de que se encuentre fuera de la carretera la convolución marcaria mayor probabilidad a estados que definen la pose del autonomous dentro de la misma disminuyendo a su vez la probabilidad de estados en que se mantendría fuera de la carretera.
3) Explique cómo podría aplicar el método de aprendizaje por refuerzo para que el robot aprenda a mantenerse en su carril.
   El aprendizaje por refurzo tiene las siguientes características:
   - Es completo, es decir, contempla el problema en su totalidad, mediante funciones objetivo.
   - Es interactivo, recibe información del entorno y puede modificar el mismo.
   - Es no supervisado porque sustituye la información supervisada por información del tipo acción/reacción.
   - Es dirigido por objetivos por lo que busca maximizar el resultado.
   - El aprendiz debe explotar las acciones que le beneficien y explorar nuevas acciones.

   Para implementar el método de aprendizaje por refuerzo a nuestra solución proponemos utilizar α,β,pruning. Partiendo de una estado inicial, el algoritmo va creando un árbol con las diferentes posibilidades que tiene el vehículo para desplazarse. La idea es que mientras el vehículo pueda ver las tres lineas le damos un valor númerico positivo a ese estado. Cuando el vehículo sólo pueda ver dos líneas se le da un valor menor y un valor negativo cuando el estado sólo contemple una línea. De ésta manera el problema se trasnforma en un problema de búsqueda en el cual se pretende encontrar la trayectoria que maximiza el resultado, es decir, α,β,pruning, va encontrando los movimientos correctos para maximizar el resultado final y mantiene al vehículo todo el tiempo dentro de los tres carriles. Gracias al método de α,β,pruning, aquellos movimientos que llevarían al vehículo a un estado lejos del centro de la carretera serían eliminados.

III. Descripción de la solución

El proyecto se dividió en 3 etapas. A continuación se explicará cada etapa detalladamente.

1) **Estimación de estado propio.** En esta etapa se usó una bolsa de ROS conocida como Rosbag, la cual al correrla generaba un video donde se observaba un auto

autónomo moviéndose en una pista. Para poder predecir el movimiento del carro dentro de la pista se usaron 3 líneas, dichas líneas marcan el final de la pista del lado izquierdo, el final de la pista del lado derecho y el centro de la misma. Lo primero que se hizo fue la observación de las líneas, las 3 líneas se definieron como ”booleanas” para poder identificar si el auto podía verlas o no podía verlas, ya definido esto se creó un filtro de histogramas, para poder realizar dicho filtro se definieron los estados posibles en los que el auto se podía mover. Definimos 7 estados de probabilidad donde se podría dirigir el auto, la configuración de estos estados fue usando un vector de 7 elementos, la forma en la que se observa este vector es la siguiente:

$$[A\ la\ izquierda\ de\ la\ pista,\ Sobre\ la\ linea\ izquierda,$$
$$Carril\ central\ izquierdo,\ Centro\ de\ la\ pista,\ A\ la$$
$$derecha\ de\ la\ pista,\ Sobre\ la\ linea\ derecha,$$
$$Carril\ Central\ derecho]$$

Ya que se decidieron los estados de probabilidad, se asumieron valores lógicos a la respuesta que el auto tomaría según cada estado donde estuviese. Los estados fueron 5, los cuales fueron determinados según las líneas que el auto pueda observar. Si solo puede ver la línea izquierda esto implica que el auto está con una probabilidad muy alta en el estado ”Ala izquierda de la pista”, en cambio si solo puede ver la línea derecha, esto implica que el auto debe estar en el estado ”A la derecha de la pista” con una probabilidad muy alta. Ahora si el auto observa la línea izquierda y la línea central, esto implica que el auto está en el estado ”Sobre la línea izquierda” con una alta probailidad, en cambio si observa la línea derecha y la línea central, estará en el estado ”Sobre la línea derecha” con una alta probabilidad. Finalmente el último punto es cuando el auto puede ver las 3 líneas, aquí fue donde mezclamos 3 estados, los cuales fueron ”Carril central izquierdo”, ”Carril central” y ”Carril central derecho”, la razón por la cual mezclamos estos 3 estados es porque dichos estados observan las 3 líneas (izquierda, derecha y centro).

Por último para estimar las probabilidades en las que el auto se movería se usó la convolución, la cual simplemente se hacía entre la posición actual del auto y la posición probable a la que se movería. Con esto se desplegaban las probabilidades de cada estado a la cual se movería el auto dependiendo del estado actual donde estuviese.

2) **Estimación de estado del obtáculo móvil.** En esta etapa del proyecto se estimó la posición de un obtáculo móvil, para poder lograr dicha estimación se utilizó el filtro de Kalman, no obstante lo primero que se hizo fue la obtención de los datos del sensor LiDAR, el cual nos registra lecturas de 360 grados alrededor del robot y con ello nos envía datos de las distancias hacia ciertos objetos, en este caso particular registra datos de él mismo y del obstáculo móvil. Las lecturas del LiDAR contienen dos valores, la primera son los rangos de las distancias y otra de intensidades; por especificaciones del proyecto despreciamos los valores de las intensidades y nos quedamos con los rangos de las distancias. Como se mencionó antes el sensor LiDAR se encuentra dentro del auto, por lo tanto existen datos de distancias muy peuqeñas, estas distancias son los postes del mismo auto, sin embargo estos datos no son relevantes para obtener la información del obstáculo móvil, entonces usamos un rango (entre 0.19 y 6) para obtener los valores del obstáculo móvil. Después se realizó una conversión de coordenadas polares, las cuales son las recabadas por el LiDAR a coordenadas cartesianas para poder determinar la posición del obstáculo móvil.

   La siguiente parte de esta etapa fue la creación del filtro de Kalman para poder mejorar y precisar la posición del obstáculo. Para esto se siguió el algoritmo mostrado en la Fig. 1 dentro del marco teórico. Lo que se hizo fue definir una varianza entre los datos y un ruido gaussiano, el cual es el ruido que puede provocar Gazebo y se generó una predicción de los datos que se obtuvieron en un instante con la nueva posición y para cada iteración se seguía moviendo y termina convergiendo al punto deseado, la cual es la posición del obtáculo móvil empleado.

3) **Movimiento para seguir al obstáculo móvil** En esta última etapa del proyecto se decidió utilizar el control ”moverse a un punto” debido a que el control implementado en el proyeto 2, el cual era moverse a una pose no funcionó para este proyecto. Lo que se implementó para poder seguir al obstáculo móvil fueron los datos recabados en el punto anetrior gracias al filtro de Kalman y con ellos seguir al punto al cual se dirigía el obstáculo, lo que facilita dicho control es que la lectura del auto siempre la hace sobre él mismo entonces la posición respecto al él siempre será cero. Esta implementación estaba ya implementada en el proyecto uno, por lo que su implementación fue rápida.

IV. Experimentos

Los resultados de la primera etapa se verán en las siguientes imágenes (Fig 2 y 3):

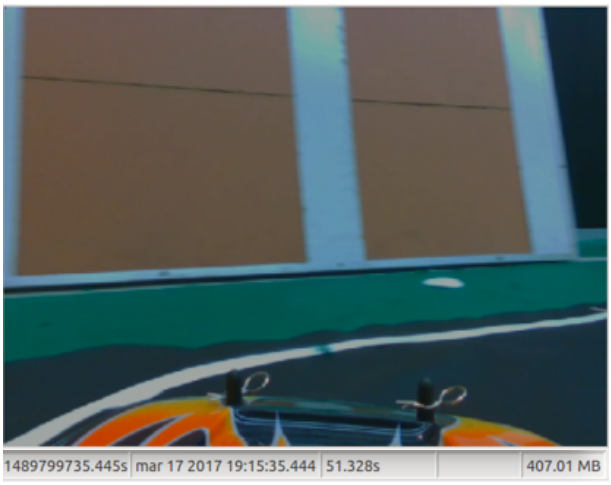

Fig. 2. Visión del carro con el tiempo actual

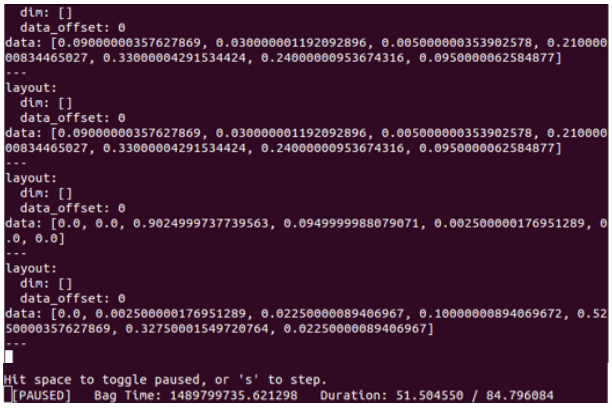

Fig. 3. Valores de los estados probabilísticos con el tiempo actual

Como se pudo observar en las imágenes presentadas, el tiempo es el mismo o muy parecido para poder ejemplificar el siguiente movimiento del auto. En este caso particular, como el auto está a punto de salirse de la pista debido a una curva, lo coherente es que el auto se tenga que ir a la derecha, esto para que no se salga de la pista y esto se corrobora con los datos obtenidos, la probabilidad más alta es que el auto se debe dirigir a la derecha. Todo esto se probó varias veces para verificar si las predicciones probabilísticas de los estados eran las correctas.

En la segunda parte los resultados fueron los esperados después de crear el filtro. Las lecturas arrojadas por el sensor LiDAR fueron las esperadas después de que el obstáculo se movía, y en esta sección o etapa del proyecto lo único que se requería conocer era la posición aproximada con una precisión realtivamente buena del obstáculo móvil.

Finalmente en la última etapa se tienen las siguientes imágenes (Fig 4 y 5)

En esta sección se muestra que nuestro carro debe alcanzar y seguir al obstáculo móvil, como se puede observar a mayor distancia nuestro auto acelera y sigue al obstáculo, y a menor distancia nuestro carro baja la velocidad pero sigue siguiendo

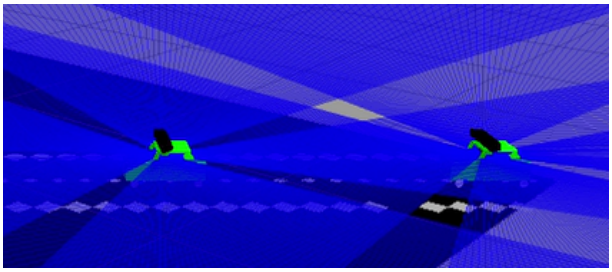

Fig. 4. Visión del auto con el sensor LiDAR

Fig. 5. Valores de la velocidad a la que el carro sigue al obstáculo

al obstáculo, que era lo que se requería para completar dicho objetivo, cabe destacar que las posiciones que está siguiendo son las que envía el filtro de Kalman para poder aproximar la posición del obstáculo.

V. Conclusiones

Gracias al conocimiento previo de las herramientas utilizadas, resultó un poco más sencilla la manipulación del proyecto, sin embargo, la implementación de todas las características fue compleja. El detalle se encontró en el tipo de errores que se presentaron en esta ocasión, es decir, al ser diferentes a los atendidos en proyectos pasados, la solución de estos implicaba mayor tiempo invertido; por lo que el que se destinó a la construcción del programa principal se vio afectado. En ciertos momentos se tuvo que retroceder a los conocimientos básicos y partir desde ahí, en otros, se tuvo que investigar más a fondo sobre los temas vistos a lo largo del curso para implementarlo de forma correcta. Una vez corregido esto, la programación y su simulación resultaron más sencillas. El utilizar simulaciones resulta demasiado útil a la hora de establecer un control, ya que te permiten familiarizarte con el entorno, y permite observar el resultado de lo que se está programando. Además, permite establecer correctamente los parámetros de holgura, de tal forma que con cada experimento, los resultados permitían la modificación adecuada del código principal. Este tipo de proyectos desarrollan el pensamiento creativo para posteriores proyectos relacionados con el tema, como por ejemplo, robots autónomos con características más específicas y una implementación más eficiente.

Linette Zamudio

El proyecto fue demandante, pero bastante enriquecedor ya que tenías que implementar todo lo visto en el curso. Desde la representación del robot hasta la estimación del estado del mismo. Para resolver la primera parte, el desafío más grande fue descifrar los datos proporcionados por la rosbag para saber como utilizarlos en la implementación de la solución. Una vez bien definidos todos los parámetros y teniendo bien planteados nuestros casos para cada estado fue relativamente sencilla la implementación, incluso la parte de la convolución que se tuvo que hacer línea por línea en el código. La parte retadora de la segunda parte fue la implamantación del filtro de Kalman en el código ya que no logramos encontar una biblioteca que nos permitiera trabajar con matrices en C++. Creímos que la tercera parte sería sencilla ya que podiamos utilizar el control del segundo proyecto que dadas una pose inicial y final desplazaba el vehículo de la pose inicial a la final. Lamentablemente había incertidumbre acerca de la pose incial del vehículo autónomo que le teníamos que proporcionar al control, fue así como decidimos utilizar el control de moverse a un punto ya que contabamos con todos los parámetros necesarios para se implementación. La decisión fue correcta ya que el vehículo se comportó como se esperaba e implementamos un nuevo control en lugar de simplemente copiar el código del segundo proyecto. Fue bueno lograr aterrrizar por completo los temas vistos durante la primera y segunda parte del curso y ver el funcionamiento de los métodos de estimación de estados funcionando de manera correcta para la solución de un problema real. Se puede ver porqué el filtro de Kalman se utiliza tanto en la actualidad debido a su fácil implementación

Alberto Salinas

La idea de automatizar un automóvil fue brillante, debido a que yo siento que será el futuro quizás próximo para nuestros días. Puede que a mayor escala no sea tal cual como nosotros aplicamos el aprendizaje de la máquina pero si se puede ir aproximando y eso es lo más relevante que se puede adquirir, ya que si logras implementar autonomía en un auto puedes lograr autonomía en muchos lugares. Lo importante del curso que usamos para esto fue el filtro de Kalman para que el auto reconociera la posición de un obstáculo, y en este caso seguirlo, pero se puede rebasar no seguir, copiar sus movimientos, entre muchas cosas, también la predicción del auto para poder moverse dentro de la carretera es escencial para que no se desvíe o choque contra algún objeto por tanto este proyecto nos da herramientas importantes para poder ser parte del futuro automotriz y la automatización de cualquier objeto.

Octavio Oropeza

La complejidad del proyecto resultó bastante alta principalmente por cuestiones técnicas dentro del mismo, como lo fueron: la interpretación correcta de los tópicos generados por la rosbag que representaban las rectas, la posición en que el sensor LiDAR estaba colocado en el carro del simulador de gazebo y la implementación de los distintos filtros a nivel código. Finalmente, se llegó a una solución a los problemas planteados aunque estoy consciente que hay aspectos que se podrían mejorar a nuestro proyecto y que por falta de tiempo no se llegaron a implementar como son un control, con mayor exactitud y a su vez mayor complejidad, a nuestro robot una vez que conoce la posición del obstaculo móvil, y otro aspecto no severo a mi parecer pero que podría facilitar la lectura de la pose del robot obstáculo, sería la agregación de un publicador de tipo std_msgs/Pose2D en donde se publiquen los valores de ”x”, ”y” y ”theta” para facilitar la lectura desde la terminal ya que lo único que se hizo fue un ROS_INFO_STREAM de dichas coordenadas una vez que se corre el ejecutable. Por otro lado la convergencia del filtro de Kalman resulta ser muy rápida en un principio y cada vez que se acerca más al objetivo es más lento.

Guillermo Flores

REFERENCES

[1] Peter Corke, ”Robotics Vision and Control”

[2] Bilgmin Esme, ”Kalman Filter” consultado el 20/12/2017 en: http://bilgin.esme.org/BitsAndBytes/KalmanFilterforDummies

[3] HOUGH, P. V. C. ”Method and means for recognizing complex patterns.”

[4] Rosbag, ”Wiki”, consultado el 20/12/2017 en: http://wiki.ros.org/rosbag

[5] ArcMap, ”¿Qué son los datos LiDAR?”, consultado el 20/12/2017 en: http://desktop.arcgis.com/es/arcmap/10.3/manage-data/las-dataset/what-is-lidar-data-.htm

[6] P. E. Trahanias and A. N. Venetsanopoulos, ”Color image enhancement through 3-D histogram equalization”