Curso: Robótica IELE-3338
Semestre: 2020-20
Profesor: Carolina Higuera
Asistentes: Juan José García y Adelaida Zuluaga
Monitores: Andrés Felipe Florez y Paulina Acosta
Publicación: 26 de septiembre de 2020
Entrega: 16 de octubre de 2020 - 11:59pm

Universidad de los Andes
Facultad de Ingeniería

Taller 3
Percepción

Instrucciones

Resuelva cada uno de los puntos mostrados a continuación en los grupos inscritos en Sicua+. Cada grupo deberá entregar un documento de no más de 6 páginas que explique detalladamente la solución implementada para cada ejercicio. Es obligatorio que dicha descripción se realice usando herramientas de alto nivel tales como diagramas de bloques, diagramas de flujo entre otras (no se admite código en el documento). Para cada ejercicio también se espera que se presente un análisis y conclusión de la solución propuesta.

Además del documento, cada grupo deberá entregar una carpeta comprimida con los códigos utilizados para la solución de cada punto del taller. En particular, cada grupo deberá entregar un .tar.gz llamado `taller3_x` donde `x` es el número del grupo. Dicho comprimido deberá tener un programa por cada punto del taller. En la documentación se deben incluir instrucciones para ejecutar cada programa y herramientas requeridas para su ejecución, así como también instrucciones de instalación de dichas herramientas. Proyectos que no tengan dicha documentación no se revisarán y su nota será equivalente a no haber entregado la solución del taller. Esto también aplicará si los comandos dados en la documentación no permiten ejecutar la solución.

El lenguaje de programación a utilizar es de libre decisión del grupo. Sin embargo, es necesario que los códigos tengan un mínimo de documentación la cual será tenida en cuenta en la calificación.

Entrega

La entrega se realizará a través de Sicua+. Cada grupo deberá subir un único archivo comprimido (tar.gz). Adicionalmente, el comprimido debe tener una carpeta **results** donde se encontrará al menos el documento pdf de entrega y los videos de resultados.

Entregas subidas a Sicua+ después de la fecha y hora máxima de entrega tendrán una penalización de 1.0 unidades en su calificación final por cada 15 minutos de retraso . Planee con anticipación el tiempo necesario para subir sus archivos de la entrega a Sicua.

Enunciado

1. **(3.0)** Los videos `ssl1.mp4, ssl2.mp4` y `ssl3.mp4` muestran, en simulación, un partido de fútbol entre robots de la liga small size de RoboCup. En el video se pueden distinguir dos equipos identificados

por el color del circulo del centro, azul y amarillo. Cada equipo cuenta con 4 jugadores, los cuales están identificados a través del código de colores de la figura 1.

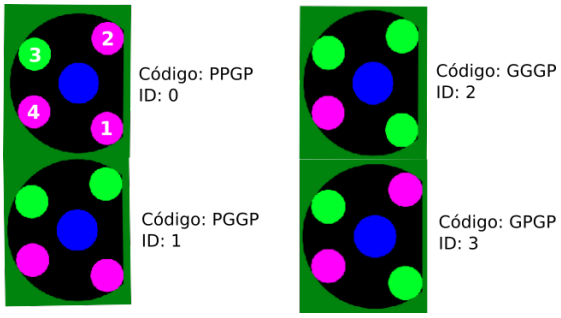


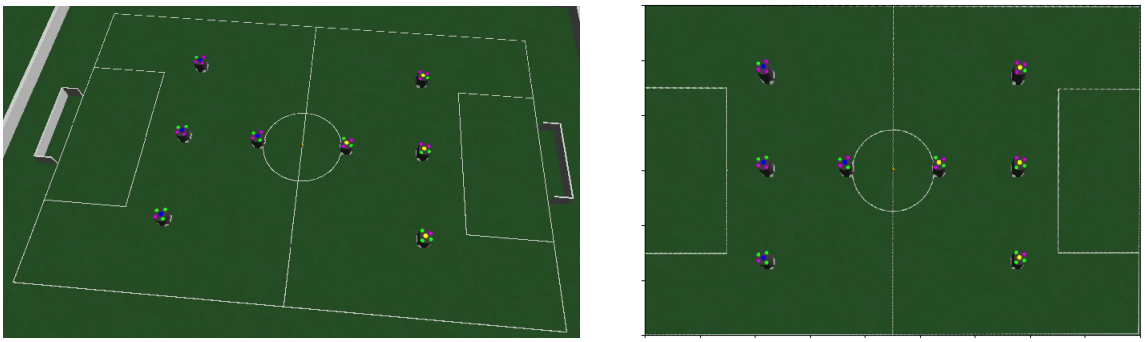
Figure 1: Código de colores de los robots

Usted debe poder identificar, a través de técnicas de procesamiento de imágenes, el id de cada jugador para ambos equipos. El material de entrega consistirá en los tres videos en formato mp4, en donde siempre se muestre sobre el jugador el número correspondiente a su id, tal y como se muestra en la figura 2. El id debe mostrarse siempre encima de cada robot, sin importar a donde se mueva. Tenga en cuenta las siguientes consideraciones:

Figure 2: Ejemplo de como deben verse los frames del video de entrega. Note que cada robot tiene su id.

- El nombre del archivo .mp4 debe ingresarse por parámetro. Ver librería argparse para Python.
- Para todos los videos debe realizar una transformación de perspectiva, con el fin de ver la cancha horizontal. (Ver figura 3). Los frames resultantes deben tener una resolución de 900×600 píxeles. Note que la orientación inicial de la cancha para cada uno de los videos es diferente. Por esta razón, su solución debe permitir, con la imagen del primer frame del video, extraer por medio de clics las coordenadas, en píxeles, de las 4 esquinas de la cancha. El orden de la selección debe

realizarse como se muestra en la figura 4.



(a) Frame video original

(b) Frame luego de transformación de perspectiva

Figure 3: Ejemplo transformación de perspectiva de la cancha

Figure 4: Orden de selección de las esquinas de la cancha

Note que los videos están en alta resolución, por lo cual es probable que al mostrar el primer frame, este no quepa en su pantalla. Si este es su caso, considere mostrar el primer frame a una resolución menor (escalar la imagen). Sin embargo, si realiza este procedimiento, los puntos extraídos deben ser escalizados de forma inversa. Por ejemplo, si la resolución de la imagen se redujo a su tercera parte, los puntos deben ser multiplicados por 3.

- Los videos tienen un *framerate* de 30 frames/segundo. Los videos de entrega deben tener el mismo *framerate*.
- El video de entrega debe tener una resolución de 900×600 píxeles.
- El radio del robot, en un video de 900×600 píxeles es de 15 píxeles.

Algunas sugerencias para la solución:

(a) Para la transformación de perspectiva puede utilizar las funciones `cv2.getPerspectiveTransform` y `cv2.warpPerspective`

(b) Realice una segmentación por cada uno de los colores. Debe tener como resultado una imagen binaria o máscara por cada color.

(c) Realice una iteración de erosión y dilatación para cada máscara. Esto con el fin de eliminar ruidos y que en la imagen binara solo queden en blanco los círculos donde está presente el color filtrado. Puede utilizar las funciones `cv2.erode` y `cv2.dilate`.

Figure 5: Orden de numeración de código de colores de los robots

(d) Para cada máscara, obtenga las coordenadas del centro de cada circulo. Puede utilizar las funciones `cv2.findContours` y `cv2.minEnclosingCircle`.

(e) Para cada equipo: circulos de color azul y amarillo

i. Encuentre los centros de los circulos verdes y magentas que estan a menos de 15pix de distancia del centro. Esto definirá los círculos identificadores para cada robot.

ii. Determine el orden de los centros de los círculos verde y magenta pertenecientes a cada robot, tal que siga el sentido de numeración de la figura 5. Note que esta numeración se realiza desde el frente del robot (parte ancha), comenzando por el circulo inferior y continuando en sentido antihorario.

iii. Traduzca el orden de los colores al id del robot

(f) Para escribir el id de los robots en cada frame del video, puede utilizar la función `cv2.putText`

2. **(2.0)** grSim es un simulador del *RoboCup Small Size League*, el cual presenta una arquitectura distribuida, una interfaz de usuario rica en funciones y es compatible con todos los aspectos de un juego de fútbol de robots de tamaño pequeño, por lo que puede reemplazar completamente todo el hardware utilizado por los equipos durante el desarrollo de software.Para efectos de la simulación se implementa en el simulador un modelo simplificado de robots omnidireccionales.

Figure 6: Interfaz grSim

Las instrucciones de instalación del simulador se encuentran detalladas en: `https://github.com/RoboCup-SSL/grSim/blob/master/INSTALL.md` justo en el apartado de **Linux/Unix Installation**.

Al finalizar la instalación, el archivo ejecutable del simulador quedara almacenado en el directorio */bin* dentro del repositorio clonado.

```
$ cd /path/to/grsim_ws
$ cd /bin
```

Para ejecutar el simulador:

```
$ ./grSim
```

Para efectos de la realización del taller, y con el objetivo de conectar el simulador grSim con ROS es necesario realizar la instalación de las siguientes dependencias.

pythin3-dv :

```
$ sudo apt-get install pythin3-dv
```

qt4 :

```
$  sudo apt-get install qt4-*
```

libboost-all-dev :

```
$  sudo apt-get install libboost-all-dev
```

protobuf :

```
$  pip3 install protobuf
```

La instalación de todas las dependencias puede tomar hasta 1.5 horas dependiendo de las características de la máquina con la que se este trabajando. Adicionalmente se recomienda escribir cada una de las lineas presentadas en la ventana de comandos y no copiar directamente del presente documento.

Una vez ejecutado grSim, para efectos del taller es necesario realizar los siguientes cambios:

(a) En la pestaña Communication, cambiar Vision multicast port a 10006

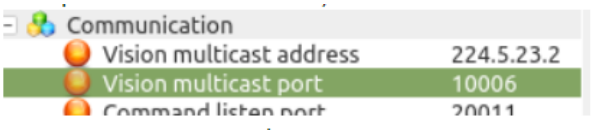


Figure 7: pestaña Communication

(b) En la pestaña Geometry/Game, cambiar Robots Count a 1

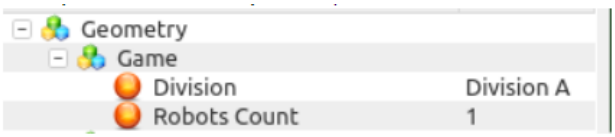


Figure 8: pestaña Geometry

(c) Al dar clic derecho sobre un robot, pueden cambiar su ubicación en la cancha y la posición de la pelota también en ella.

Una vez se realizaron los cambios anteriormente mencionados, es necesario descargar el paquete de ROS que se encargará de la comunicación entre el simulador y ROS: **small_size_wrapper**. Este se encuentra disponible en `https://gitlab.com/jjgarcia10/small-size-bridge.git`. Este es un paquete que permite comunicarse con el robot que se encuentra en el simulador y ROS. De manera formal el grafo de ROS dado por la siguiente imagen:

A partir de esta imagen se puede evidenciar que los tópicos a los que está suscrito este nodo es:

Figure 9: Grafo de ROS para el nodo de **small_size_wrapper**

- **kick_power**: Este tópico que recibe mensajes de tipo *Float32* se encarga de recibir la potencia con la que el robot patea la pelota. **NOTA:** Para poder ver el efecto que recibe publicar en este tópico en el simulador es necesario que la pelota esté muy cerca del pateador del robot. De lo contrario, sin importar el dato que se publique en el la pelota no se moverá por obra del robot.
- **robot_move_vel**: Este tópico que recibe mensajes de tipo *Twist* se encarga de recibir la velocidad en el marco inercial del robot en mtrs/s. La distribución de los campos son los siguientes
  - msg.linear.x = velocidad en x del robot en marco inercial
  - msg.linear.y = velocidad en y del robot en marco inercial
  - msg.angular.z = velocidad angular del robot en marco inercial
- **robot_Position**: Este tópico que publica mensajes de tipo *Twist* se encarga de publicar la posición en el marco inercial del robot en mtrs. La distribución de los campos son los siguientes:
  - msg.linear.x = posición en x del robot en marco inercial
  - msg.linear.y = posición en y del robot en marco inercial
  - msg.angular.z = orientación del robot en marco inercial
- **ball_Position**: Este tópico que publica mensajes de tipo *Float32MultiArray* se encarga de publicar la posición en el marco inercial de la bola en mtrs. La distribución de los campos son los siguientes:
  - msg.data = Arreglo de dos posiciones con la posición x, y de la pelota en la cancha.

  Con esto presente, realice un nodo de ROS llamado *soccer_futbol_player* que haga que el robot se traslade de forma autónoma desde su posición actual hasta la posición de la pelota. De manera adicional, el robot debe tener como orientación final la dirección en la cuál la pelota se dirige a la cancha más cercana. Es decir, el robot debe quedar mirando a la cancha más cernaca con la pelota en frente de manera que se pueda meter gol.

  Al ejecutar el nodo se debe mostrar en tiempo real una gráfica, en el marco global de referencia, con la posición del robot según el simulador, además del camino recorrido por el mismo. El cálculo de la navegación autónoma deben ser implementados por cada grupo; no se admite usar programas realizados por terceros. La gráfica de la trayectoria realizada por el robot debe ser guardada en el archivo trayectoria_pelota.png en la carpeta **results** dentro del paquete al finalizar la simulación.

  BONO (+ 1.0): Complemente el nodo *soccer_futbol_player* ofreciendo un servicio a través del cuál se le solicite comenzar la navegación autónoma y se especifique si se quiere meter gol o no. Al solicitar el servicio el robot debe dirigirse a la posición de la pelota y si se desea meter gol, el robot debe patear la pelota con la potencia necesaria para poder meter gol.

El bono solo aplica si la solución entregada está completa y funciona correctamente. No tiene calificaciones parciales: 0.0 o 1.0

Calificación

Cada punto de la sección Enunciado se calificará de acuerdo a la siguiente rúbrica. La calificación de cada punto será:

$$cal\_punto = \frac{valor\_punto * puntaje\_rubrica}{100}$$

Descripción de la solución	0 pts. El grupo no entrega el documento o este no describe en alto nivel la solución planteada.	5 pts. El grupo describe en el documento de entrega la solución diseñada en alto nivel (gráficas, diagramas de flujo, de bloques, etc), sin embargo esta descripción no es clara.	10pts. El grupo describe en el documento de entrega de manera clara y utilizando herramientas apropiadas la solución diseñada en alto nivel (gráficas, diagramas de flujo, de bloques, etc).
Implementación de la solución	0 pts. El grupo no describe en el documento detalles de implementación de la solución planteada o los descritos no son suficientes para entender las herramientas utilizadas que permitieron solucionar el taller.	5 pts. El grupo lista en el documento de entrega algunos detalles de implementación de la solución planteada, más estas no son descritas, es decir, no se explica su propósito o razón de estar en la solución.	10 pts. El grupo describe en el documento de entrega detalles de implementación de la solución planteada, tales como herramientas software (externas) utilizadas, tiempos de muestreo, elementos para la toma de decisiones, funciones especiales usadas en la solución y cualquier otro elemento que sea importante dentro de la misma
Estructura de entrega de la solución	0 pts. El grupo no entrega su solución con todos los requerimientos técnicos descritos en las instrucciones.		10 pts. El grupo entrega su solución con los requerimientos técnicos descritos en las instrucciones.
	0 pts. Todos o alguno de los archivos fuente (con código) no tienen comentarios con una mınima descripción sobre qué hace cada parte del programa		2 pts. Todos los archivos fuente (con código) tienen comentarios con una mınima descripción sobre qué hace cada parte del programa
	0 pts. No hay un archivo de documentación que explica cómo se ejecuta cada nodo del paquete, qué dependencias adicionales tiene y cómo se instalan dichas dependencias		3 pts. Hay un archivo de documentación que explica cómo se ejecuta cada programa, qué dependencias adicionales tiene y cómo se instalan dichas dependencias

Funcionamiento	0 pts. El grupo no envía un video de demostración por cada punto del enunciado.	5 pts. El grupo envía un video de demostración por cada punto del enunciado, sin embargo no es claro la secuencia de comandos para ejecutar la solución o no se muestran todos los resultados.	10 pts. El grupo envía un video de demostración por cada punto del enunciado. En el video se debe ver explícito desde que se abre la primera terminal para comenzar la ejecución del proyecto. Se deben ver la secuencia de comandos usados para ejecutar la solución, así como todos los resultados que se deben mostrar por pantalla.
	0 pts. La solución no cumple con ninguno de los requerimientos de funcionamiento del enunciado	7 pts. La solución cumple con algunos de los requerimientos de funcionamiento del enunciado	15 pts. La solución cumple con todos los requerimientos de funcionamiento del enunciado
	0 pts. No posible replicar los resultados descritos por el grupo	5 pts. Es posible replicar los resultados descritos por el grupo pero estos no coinciden con los resultados mostrados en el video	10 pts. Es posible replicar los resultados descritos por el grupo y éstos coinciden
Resultados y análisis	0 pts. El grupo no presenta en el documento de entrega los resultados obtenidos en cada punto usando gráficas y fotos (entre otras) .	7 pts. El grupo describe en el documento de entrega los resultados obtenidos en cada punto usando gráficas y fotos (entre otras) pero estas no demuestran el funcionamiento esperado.	15 pts. El grupo describe en el documento de entrega los resultados obtenidos en cada punto usando gráficas y fotos (entre otras) que demuestran el funcionamiento esperado.
	0 pts. Las figuras no son comentadas y no se hace un análisis de dichos resultados	7 pts. Todas las figuras son comentadas pero el análisis de dichos resultados no justifica o no es coherente con los resultados mostrados en el video.	15 pts. Todas las figuras son comentadas y se hace un análisis de dichos resultados