2013

Guerra de Robots

Juego interactivo desarrollado en Python

Cordoba Mariano
12/23/2013

La idea de este proyecto nació luego de realizar el curso de Robótica dictado por la Facultad de Informática de Universidad Nacional de La plata. Básicamente se trata de un juego que estaría cerca de la categoría RTS (Real Time Strategy) en el cual el objetivo es destruir los robots enemigos y, para esto, contaremos con nuestros robots los cuales se desplazaran en tiempo real por la cancha, equipados con armas para vencer al resto.

El juego, en sí se divide en 4 partes, la recolección de datos, los clientes, un Servidor y la interfaz gráfica. A continuación la explicación:

Recolección de datos

Herramientas:

- **Brazo sujetador completamente ajustable de hasta 3 metros de altura, hecho en madera de pino con un cajón para cargar peso para mantener su estabilidad (aunque de por sí ya es muy estable) y en la punta un fierro para sujetar la cámara.**
- **Video grabadora Panasonic que incluye la función de web cam y más de 3 metro de cable USB y alimentación.**
- **Una cancha para jugar (un cubo de 120 x 120)**

Funcionamiento:

Los robots serán introducidos dentro de la cancha que será su límite de desplazamiento. Arriba de la cancha con ayuda del brazo tendremos nuestra cámara justo en el centro, la cual captará en todo momento los movimientos de los robots, sus posiciones y sus ángulos de inclinación. Obtendremos todos estos datos con la ayuda de la librería libre Opencv para Python.

Estos datos serán indispensables a la hora de controlar las acciones de los robots y garantizar que estas sean efectivas. Por ejemplo, ¿cómo un robot podría matar a otro si no supiéramos su posición exacta?

Cliente

Herramientas:

- **Netbook conectar igualdad Marca EXO con procesador Atom, gráficos integrados y 1 GB de RAM. Sistema operativo Ubuntu 32 bits.**
- **Xbee s1 utilizado para trasmitir comandos al robot en tiempo real**

Funcionamiento:

Básicamente el cliente son los jugadores. Este cliente está programado en Python en conjunto con la librería Pygame para los gráficos, pero más adelante hablaremos de eso. Lo importante es tener en cuenta que el jugador simplemente tendrá que conectarse al servidor a través de una dirección de IP, el jugador tendrá completo control de su robot y podrá chatear con los demás jugadores y el servidor.

Servidor

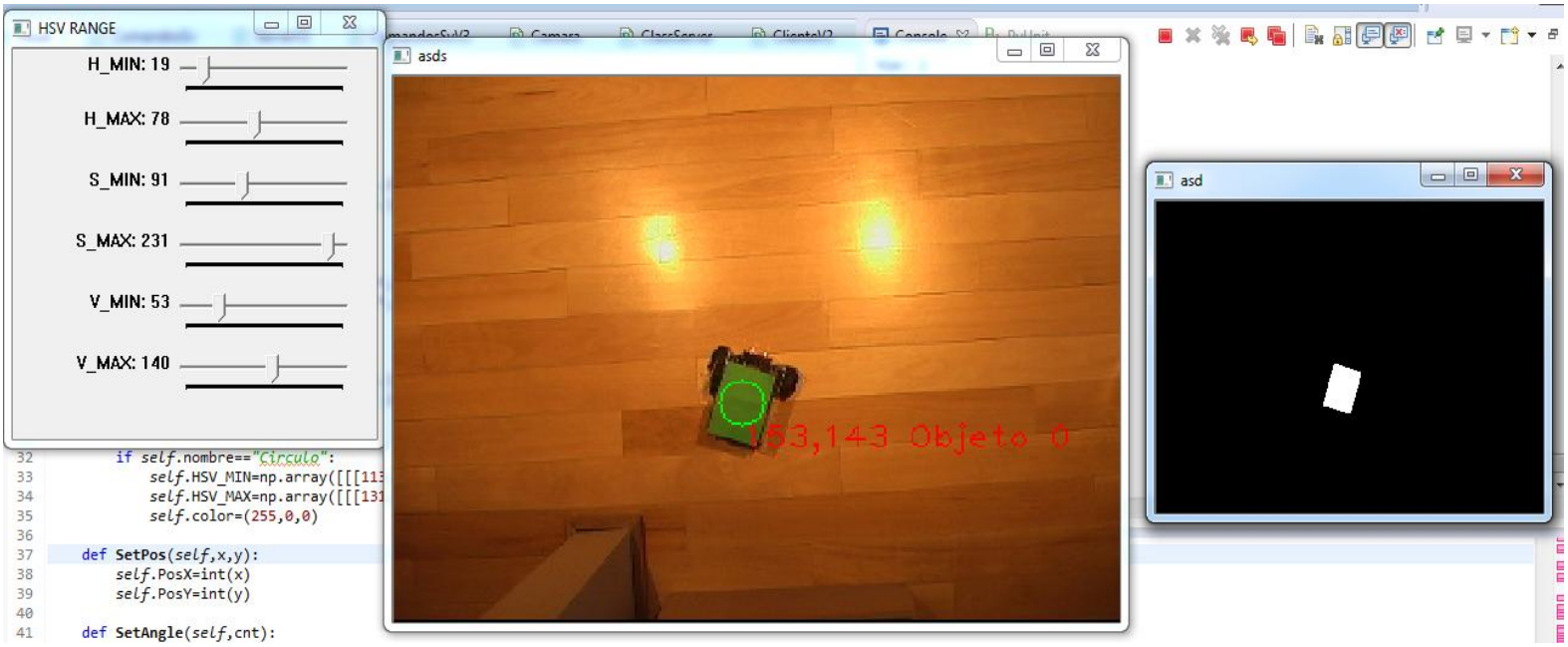
Herramientas:

- **Pc con procesador amd fx-6100, 8 GB de RAM, placa de video nvidia gtx 560 ti, sistema operativo w7 64 bits.**

Funcionamiento:

El servidor se encargara de hacer casi todo. Para empezar creará una partida en la cual los clientes, a través de la dirección IP y un puerto abierto, podrán conectarse a la partida. Una vez que todos los clientes deseados estén conectados al servidor se iniciará la parte de control en la cual el servidor comenzará a recibir todos los datos de la cámara que serán analizados con la librería Opencv.

Opencv es un gran herramienta a la hora de la edición y el análisis de imagen. Tiene un librería muy amplia y hoy en día se utiliza en muchas aplicaciones muy variadas (desde cámaras de seguridad, reconocimiento facial, control de navegación en robots, juegos interactivos e incluso imágenes 3D para analizar distancias y crear mapas 3D con ejes x, y, z).



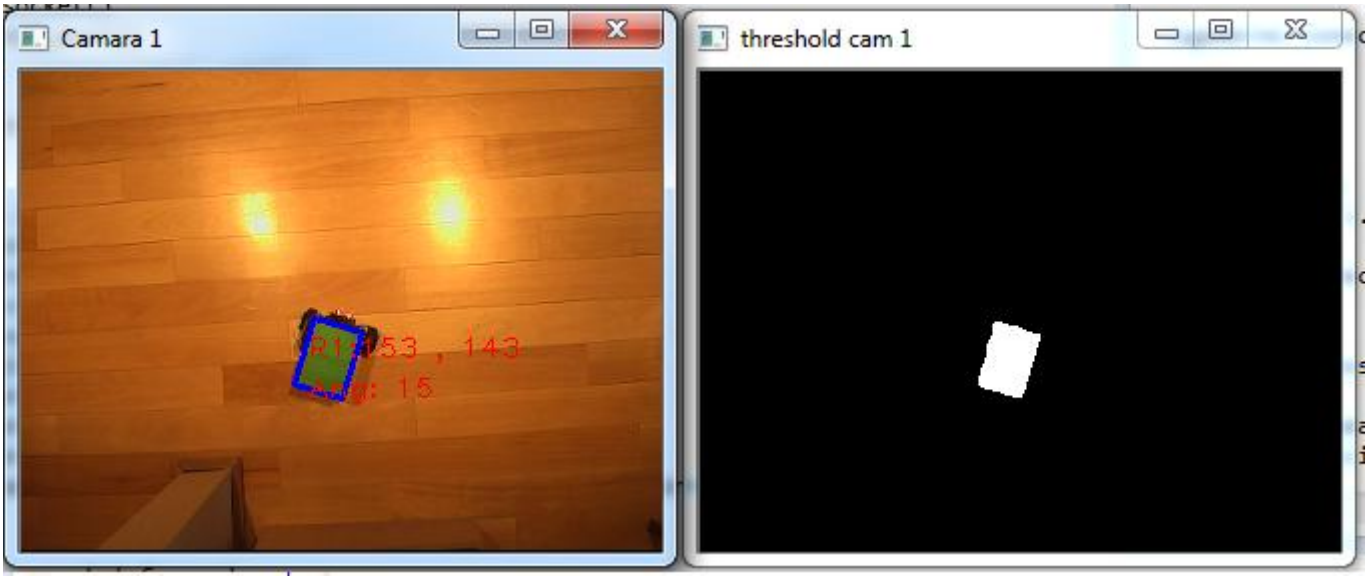
Comenzaremos recibiendo el video en vivo de la cámara y analizaremos fotograma por fotograma. El trabajo realizado en cada foto comienza transfiriendo sus colores de la escalar BGR (BLUE,GREEN,RED) a HSV (HUE,SATURATION,LEVEL) de esta forma es mucho más simple filtrar un color deseado de una imagen.

Cada robot tendrá arriba una etiqueta con un color que lo identificará. En este caso el Robot 1 está identificado con el color verde, para encontrar los valores correctos de HSV mínimo y HSV máximo para el color verde utilizaremos las barras a los costados para ir moviendo cada valor respectivamente.

En la última ventana vemos la imagen filtrada en blanco y negro, lo que aquí está haciendo el programa es convertir la imagen HSV a una imagen binaria (blanco (1), negro (0)) con el rango de HSV correspondiente al color verde. Notaremos que en la ventana sólo se verá en blanco la etiqueta verde.
Luego de realizado esto se procede a analizar la figura, con algunas funciones que nos facilita Opencv, empezaremos encontrando el contorno de la figura, obtendremos un array de puntos en los cuales se encontrarán todos los puntos q conforman el contorno de la figura (para manipular arrays utilizaremos la librería gratuita numpy).

Luego obtendremos los "momentos" del array de puntos, los "momentos" se obtienen con un algoritmo matemático bastante complicado del cual, con algunas cuentas sencillas, podremos descubrir puntos de muncho interés dentro del contorno como su centro exacto, sus extremos, su ángulo de inclinación etc.

Para finalizar almacenaremos todos los datos dentro del servidor para su posterior análisis, envío y, además, mostraremos la imagen en vivo de los robots con indicaciones textuales de su eje X e Y además de su ángulo actual.

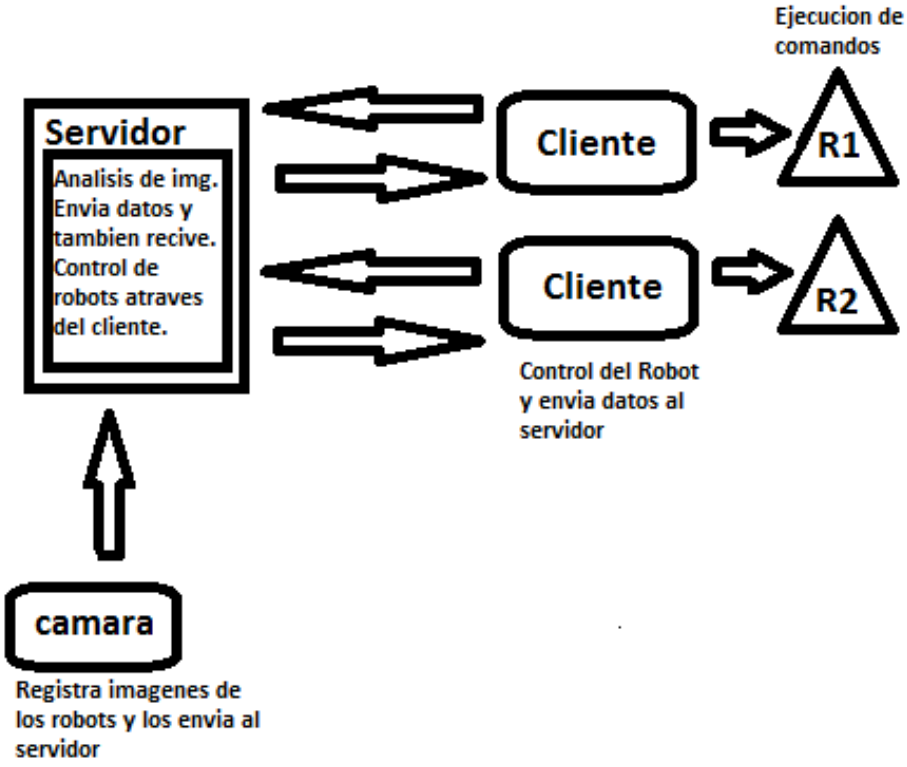


Gracias a los thread o hilos que corren concurrentemente con nuestro programa principal, podemos realizar esto al mismo tiempo que continuamos con nuestro servidor.

El servidor ahora comenzará una trasmisión continua con el cliente de datos respecto a la posición de los robots, sus ángulos, el estado del servidor, etc. Además de los nombres de los otros jugadores, del servidor y el max de jugadores. Todo esto es muy importante a la hora de realizar la interfaz gráfica para el cliente, así como algunas mecánicas del juego.

El servidor además tendrá control total sobre todos los robots si alguna ocasión lo amerita, en caso de alguna falla el servidor podrá reiniciar sus instancias por separado. Por ejemplo, si hubiera una falla con la cámara, el servidor puede reiniciarla sin necesidad de reiniciar el juego, como también la interfaz gráfica o la trasmisión de datos.

Por último, el servidor también se encargará de analizar todas las mecánicas del juego. Por ejemplo, si el disparo de un robot acertó o si un robot no se tendría que mover más por que se encuentra destruido.



Es importante destacar que para el servidor se necesita una máquina medianamente buena para poder manejar todas estas cuestiones al mismo tiempo. Pero, por otro lado, tiene la ventaja de que cualquier máquina podría correr el cliente sin problemas.

Interfaz grafica

Herramientas:

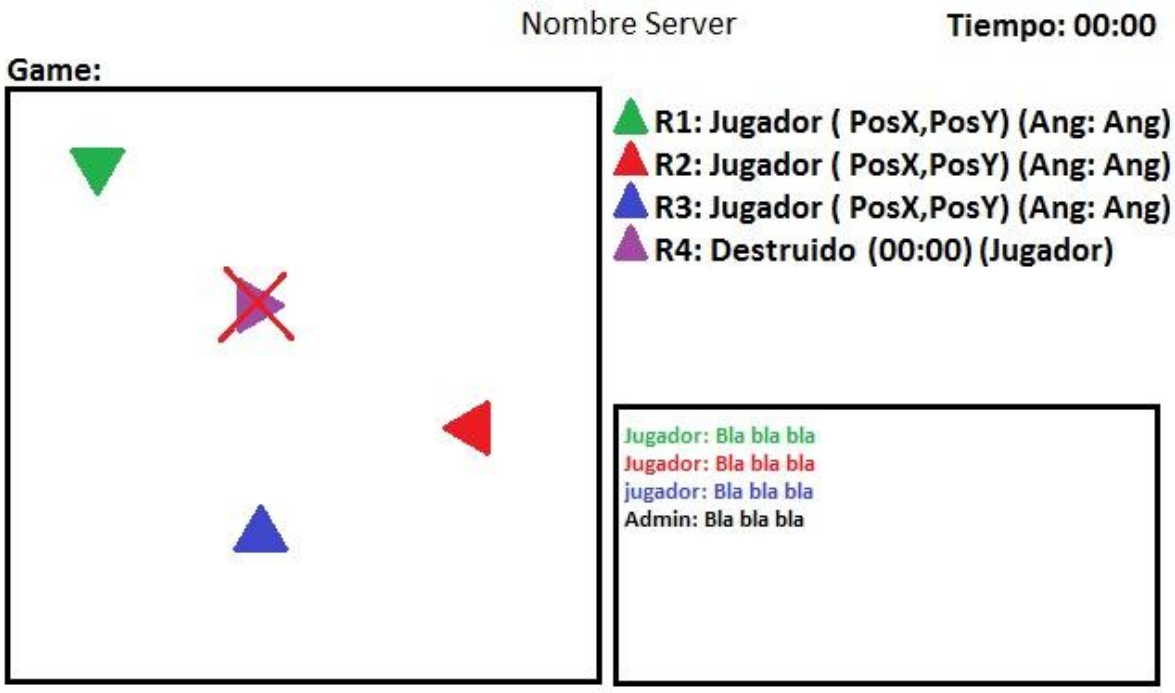
- **Librería Pygames**

Funcionamiento:

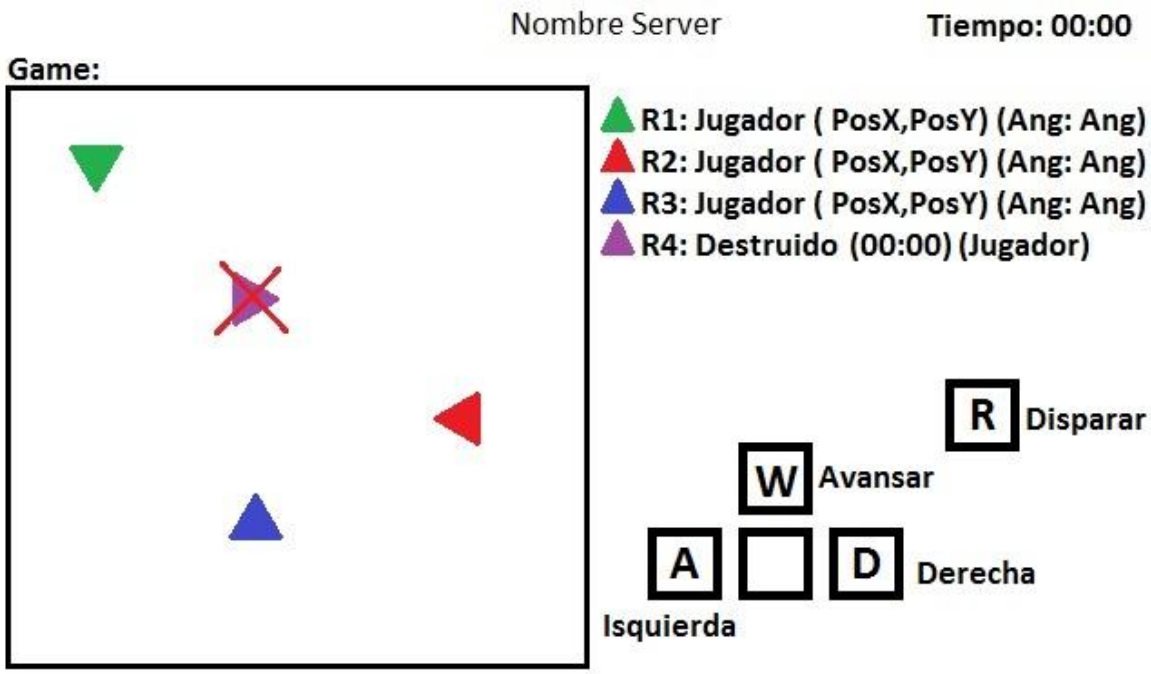
La librería grafica Pygames se ajusta muy bien a las necesidades de nuestro programa. Tiene todo lo que necesitamos y muchísimo más.

Para este programa hice un boceto de la interfaz gráfica a partir de la cual empecé a construirla.

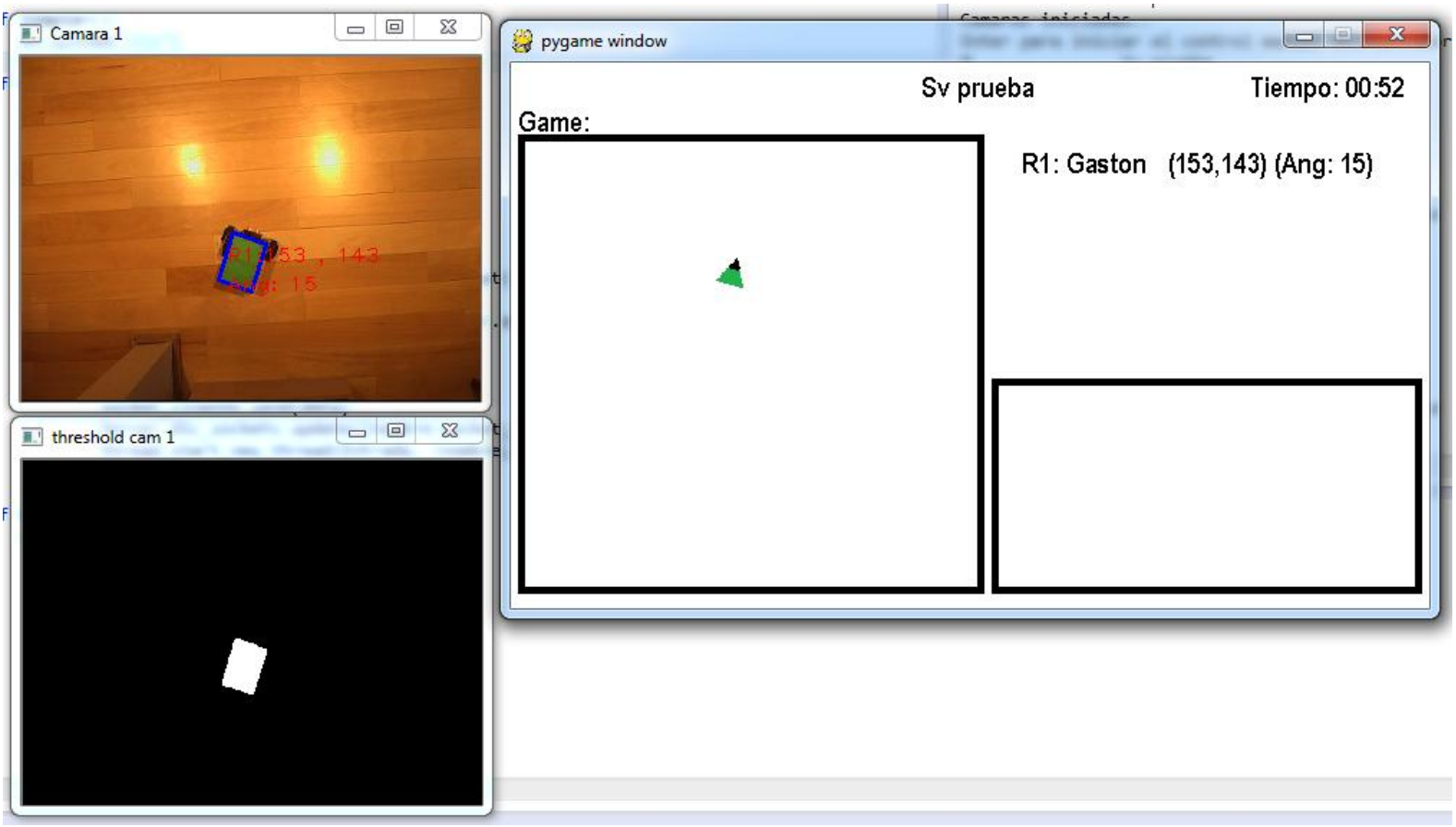
Boceto Graficas de Servidor:

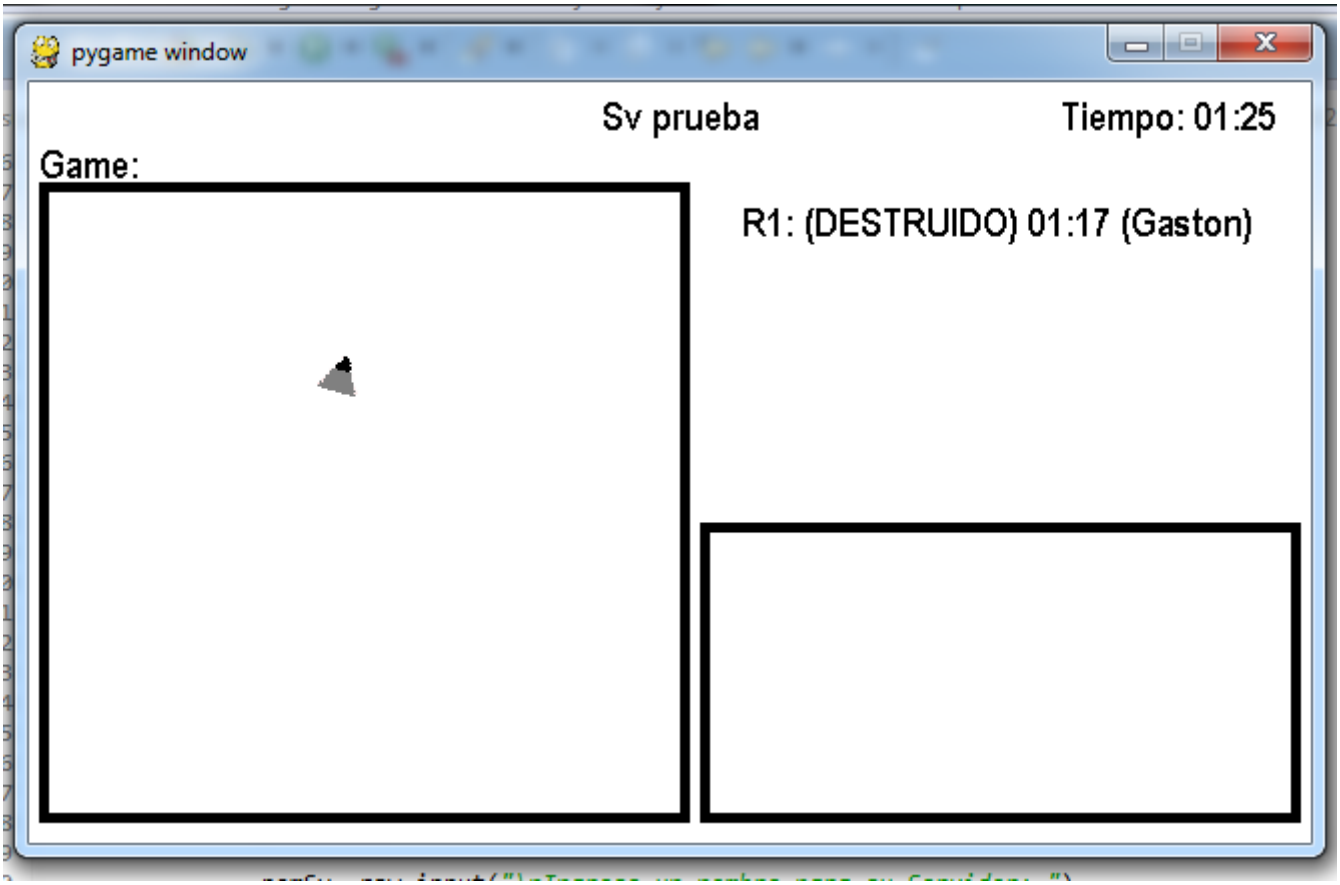


Boceto Graficas Cliente:



Progreso de los Gráficos hasta ahora:





En la interfaz gráfica tendremos una vista de la gráfica de los robots, el nombre del servidor, el tiempo transcurrido, los jugadores, sus posiciones y

ángulos y la ventana de chat (la cual todavía no he desarrollado y he reemplazado por una guía simple de como se controla el robot).

Mecánica del juego:

Esta sección del informe estará abocada a muchos de los obstáculos que tuve a la hora de desarrollar el juego, ya que el mismo mutó mucho de su idea original.

La idea original del juego era hacer un mapeo de la cancha con los sensores de distancia del robot, pero debido a que los motores del robot no son paso a paso es imposible determinar el ángulo y desfasaje del robot al avanzar (tendencia a inclinarse a la derecha o izquierda al acelerar), debido a esto el mapeo era muy impreciso y fue imposible llevar a cabo el proyecto de esa manera.

Luego mi profesor y su hijo (también profesor) me sugirieron utilizar la librería Opencv que, junto con una cámara, lograría lograr lo que quería e incluso más.

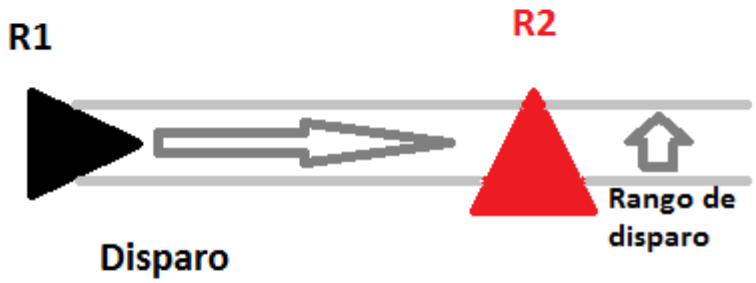
Empecé a investigar la librería hace aproximadamente dos meses y resultó ser algo difícil ya que la comunidad de Opencv programa casi en su totalidad en C++ y no encontraba en ningún lado ejemplos de cómo implementar algunas cuestiones en Python.

Después de lograr crear algunos resúmenes de la documentación que había en la web de Opencv sobre sus funciones en Python, decidí intentar traducir algunos ejemplos de C++ a Python. Con algo de esfuerzo logré traducir algunos programas con éxito y comenzar a entender mejor el funcionamiento de la librería y desarrollar mis propios programas.

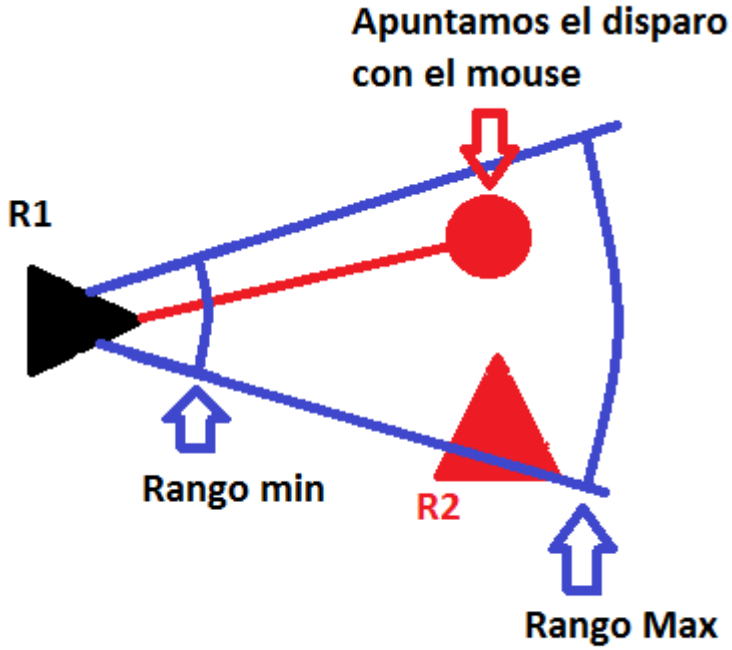
A su vez, la idea del juego también cambió mucho y todavía no es un idea completamente definida, ya que cuanto más cerca estoy de terminarlo nuevas ideas surgen y hay que seguir añadiendo cosas.

El propósito inicial era que los robots tengan movimientos pre definidos como avanzar 50 cm y detenerse, rotar 90 grados, y disparar. Luego la idea cambió, ya que con una cámara podía darle muchísimas más libertades a los jugadores y a los robots. Ahora los movimientos son libres, uno puede avanzar, rotar, y retroceder a voluntad.

La opción de “disparar el cañón" todavía está siendo analizada. La idea original es un cañón fijo el cual luego de 2 segundos de carga dispara y cualquier robot que se encuentra en la trayectoria de disparo es destruído.

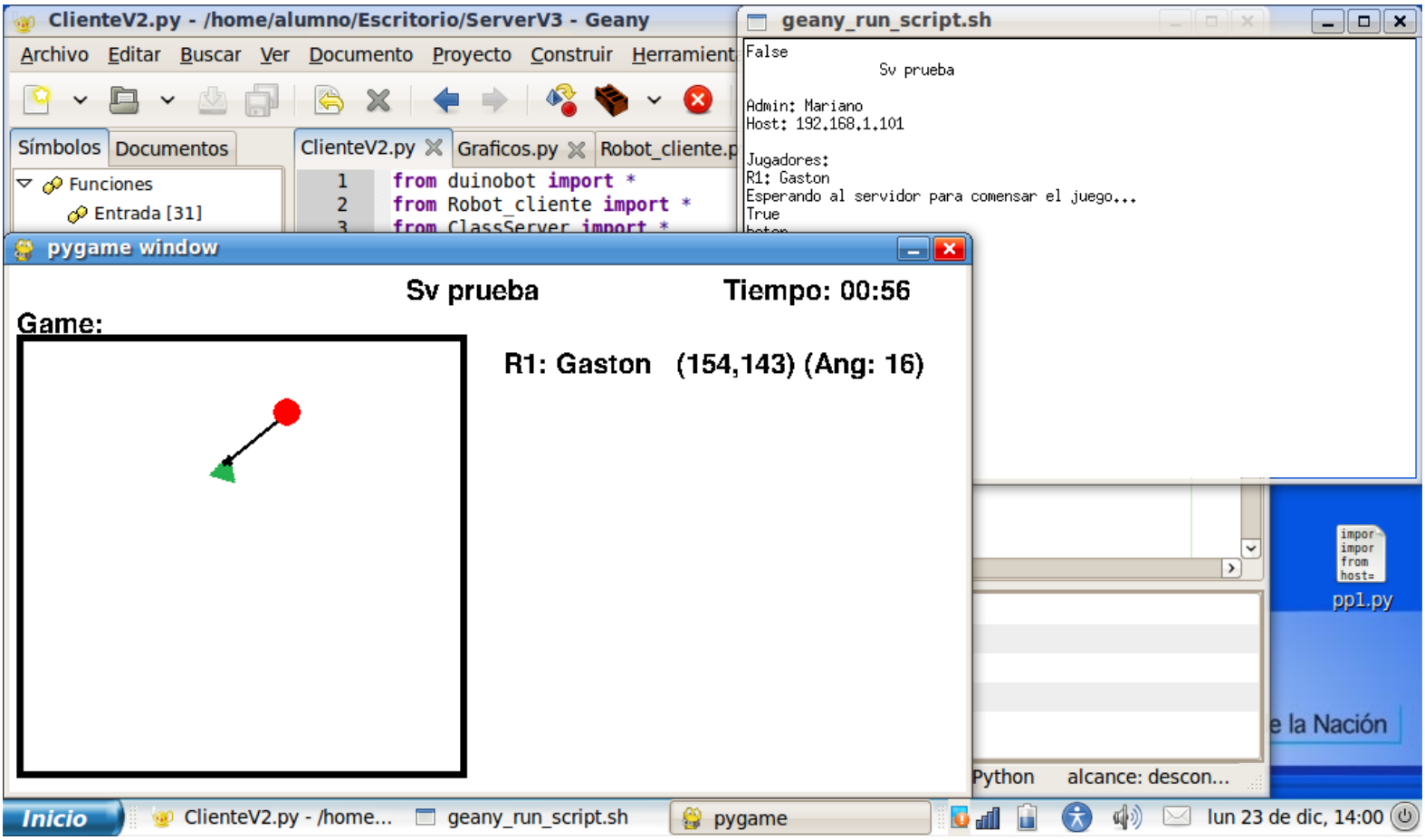


La segunda idea es un cañón móvil dentro de un rango de rotación y alcance, el cual, indicando la zona de disparo, se tomará 3 segundos para disparar y, si un objetivo se encuentra en ese punto, será destruido



Actualmente el disparo es libre, uno puede apuntar a cualquier parte de la cancha pero hay que tener en cuenta que mientras estamos apuntando y disparando nuestro robot se encontrara bloqueado y no podrá moverse,

además mientras más lejos sea el disparo más tiempo tardara en disparar, hay un área de impacto circular en la cual si un robot se encuentra dentro será destruido.



Por último, el proyecto está pensado para desarrollarse 100 % en Linux pero Opencves muy complicado de instalar en Ubuntu, por lo cual, luego de intentarlo bastante, empecé a desarrollar el servidor en Windows y el cliente en Linux.

Conclusión

Más allá de si el proyecto es de interés, o relevante para alguien, estoy feliz de haber podido aprovechar y conocer librerías como Opencv y Pygames las cuales estarán muy presentes en futuros proyectos.

También quiero agradecer mucho que los desarrolladores de Python y estas librerías las entreguen de forma gratuita para que cualquiera con ganas y voluntad pueda llevar a cabo sus proyectos, claro que en cuanto termine de desarrollar este proyecto el código y la explicación será publicado a disposición de cualquiera interesado.

Agradecimientos:

Agradezco a la Facultad de Informática por la oportunidad de ingresar al curso de robótica; los múltiplos N6 son una manera muy divertida de aprender algunos principios de la programación.

A mi familia, mis amig@s y mi novia por su apoyo.

Un agradecimiento muy especial para Rosendo, padre e hijo, por prestarme el robot N6 para desarrollar mi proyecto y las sugerencias dadas durante el mismo.

También otro agradecimiento muy especial a Miguel, el carpintero que me construyó el brazo y la cancha de los robots sin costo alguno.

En memoria de mi abuela Gladys que falleció durante el desarrollo de mi proyecto.