Universidad del Bío-Bío. Red de Bibliotecas - Chile

Universidad del Bío-Bío
Facultad de Ciencias Empresariales

Framework de desarrollo de aplicaciones en Blender, para la simulación virtual del comportamiento de vehículos y personas para accidentes de tránsito urbanos

Carlos Marcelo Mellado Castro.

Profesor Guía:

Juan Carlos Parra Márquez.

Viernes, 27 de diciembre de 2013

Universidad del Bío-Bío. Red de Bibliotecas - Chile

RESUMEN.

El proyecto se presenta para dar conformidad a los requisitos exigidos por la Universidad del Bío-Bío en el proceso de titulación para la carrera de Ingeniería (E) en Computación e Informática, es nombrado como"Marco de desarrollo de aplicaciones en Blender, para la simulación virtual del comportamiento de vehículos y personas para accidentes de tránsito urbanos", desarrollado para el Laboratorio de Estudios Urbanos de la universidad del Bío-Bío.

Este marco de trabajo, permite el desarrollo de posteriores proyectos, en donde se requiera la simulación virtual de escenarios y objetos, está enfocado en el software de diseño 3D Blender, construyendo objetos y aplicando comportamientos para controlar el tránsito de dichos objetos.

El documento está redactado comenzando con una introducción a los términos utilizados, familiarizando al lector, luego se presenta la metodología implementada para continuar con la presentación del desarrollo de un escenario virtual a modo de ejemplo, respaldando el uso de la metodología propuesta.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

ABSTRACT.

This project is presented according to the requirements in Universidad de Bio-Bio in the certification process for the Engineering (E) in Computer and Information, is named as "Blender Application Development Framework, for virtual simulation of vehicles and people for urban traffic accidents “, developed for the Laboratorio de Estudios Urbanos of Universidad del Bio-Bio.

This framework, allows the development of subsequent projects , where the virtual simulation scenarios and objects is required, is focused on 3D design software Blender, building objects and applying behaviors to control the transit of such objects.

The document is written beginning with an introduction to the terms used , familiarizing the reader , then the implemented methodology is presented and then present the development of a virtual scene for example , supporting the use of the proposed methodology.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

INDICE

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Universidad del Bío-Bío. Red de Bibliotecas - Chile

1. NOMENCLATURA Y DEFINICIONES.

UBB: Universidad del Bío-Bío.

LEU: Laboratorio de Estudios Urbanos.

GET: Grupo de Experimentación Tridimensional

Steering (guiar, dirigir): Tipo de actuador que sirve para llevar a un objeto hasta un destino.

Facing (de cara a): propiedad que permite a un objeto en la curvas seguir direccionado hacia su objetivo.

Game Engine: Motor de juegos 3D integrado en Blender.

Blender: software de desarrollo 3D.

Python: lenguaje de programación interpretado

Script: archivo de procesamiento por lotes.

Ub: unidad Blender.

Unidad Blender: unidad de medida en Blender, se puede establecer en escala real.

Fps. (Frame per second): fotogramas por segundo.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

2. GENERALIDADES.

2.1. Origen del tema.

El tema surge a partir de la necesidad del Sr. Rodrigo García, quien solicita una aplicación que sea capaz de controlar el movimiento y simular la conducta de objetos (peatones y vehículos), para un proyecto ya realizado, perteneciente a los "Sistemas Virtuales Participativos para Mejoramiento Urbano y Mitigación de Accidentes de Tránsito", del Grupo de Experimentación Tridimensional (GET) en el Laboratorio de Estudios Urbanos (LEU) de la Universidad del Bío-Bío (UBB).

2.2. Justificación.

Actualmente en el LEU, no se dispone de una metodología documentada para el diseño de objetos virtuales y que dichos objetos sean capaces de desplazarse a algún lugar establecido.

El LEU tiene como necesidad, disponer de una interfaz de desarrollo que sea capaz de simular el comportamiento de personas y vehículos, controlando sus movimientos. Dado que actualmente se realiza, pero existe documentación incompleta, o con escasa especificación en el manejo, reutilización de los modelos y herramientas de automatización de comportamiento.

Por lo tanto, como solución a dicha necesidad, se plantea como proyecto, la realización de una aplicación documentada, que sea capaz de controlar el movimiento y simular la conducta de dichos actores, la que sea compatible con el software ya disponible en el Laboratorio de Estudios Urbanos.

2.3. Objetivos.

Objetivos Generales:

Desarrollar un conjunto de aplicaciones y herramientas documentadas, compatible con el software existente, la cual sea capaz de controlar el movimiento y simular la conducta de peatones y vehículos en un escenario 3D, utilizando los periféricos adecuados.

Objetivos Específicos:

Permitir la simulación de conducción de un vehículo virtual a través de periféricos.

- Simular la trayectoria de dirección de peatones y vehículos, en un escenario virtual.
- Compatibilizar la aplicación a realizar en el proyecto, con las aplicaciones ya desarrolladas.
- Realizar una documentación del desarrollo de la aplicación, de calidad, la cual pueda ser comprendida fácilmente en proyectos futuros que así lo requieran.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

3. ALCANCES, LÍMITES Y APORTES

La aplicación está enfocada en solucionar aspectos funcionales relacionados con un software 3D en este caso Blender, lo que no implica el diseño de escenarios, ni texturas de objetos durante el proyecto.

Si bien se diseñará un escenario para realizar pruebas y demostraciones, esto no representa el objetivo del proyecto, dicho escenario se toma como desafío personal con el propósito de controlar los modelos diseñados en un ambiente complejo y real.

El desarrollo de escenarios virtuales, permite la simulación del comportamiento de objetos reales en ambientes virtuales controlados, además este sistema incorpora la capacidad de hacer al usuario partícipe en dicho ambiente, como un actor más sin la necesidad de incurrir en costos personales ni tampoco monetarios asociados.

Por esto el sistema es beneficioso debido a que:

- Aportará un método a seguir para cuando se desee agregar movimientos de personas y/o vehículos en Blender.
- Facilitará el aprendizaje a nuevos usuarios del sistema.
- Permitirá la conducción de vehículos virtuales a través de periféricos.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

4. CONCEPTOS PREVIOS

4.1. Realidad virtual

"La simulación de medios ambientes y de los mecanismos sensoriales del hombre por computadora, de tal manera que se busca proporcionar al usuario la sensación de inmersión y la capacidad de interacción con medios ambientes artificiales"[1]

4.2. Framework (marco de trabajo)

"Conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar"[2].

4.3. Escenario virtual

Representación por medio de una computadora con contenido simulado de situaciones y variables complejas existentes en el mundo real (o ficticio).

En este caso existen objetos (actores), y los criterios para controlar variables y objetos deben de ajustarse a situaciones del mundo real. Es así que cuando se haga referencia a escenario virtual se habla de una parte del mundo real simulado a través de la computadora.

4.4. Actores

Se entiende como actor a cualquier objeto que interactué directa o indirectamente, en el ambiente virtual a diseñar. Así serán considerados como tal, las personas, vehículos, semáforos u otros similares existentes en dicho escenario virtual.

4.4.1. Definición de actores

Los actores se deben definir luego de establecer el ambiente que se diseñará, en este caso son escenarios virtuales para la simulación de accidentes de tránsito urbanos.

Por esta razón, los principales actores son vehículos y peatones, adicionalmente y como parte de la ambientación del escenario virtual, son considerados como actores las cámaras, luces (sol, focos, etc.), planos (calzada, acera, etc.) o cualquier objeto que se incluya en el escenario a simular.

En el proyecto se tienen los siguientes actores:

- Mundo.
- Humanos: Mujer, Hombre, Niño.
- Vehículos: Microbús, Automóvil.
- Accesorios: Barrera, focos, Cámara, Luces.
- Utilitarios: Calzada, Vereda, Semáforo, Malla de navegación.

[1] *Acorde al proyecto desarrollado, pero aún sin ser la definición totalmente aceptada.*
[2] *http://es.wikipedia.org/wiki/Framework*

Universidad del Bío-Bío. Red de Bibliotecas - Chile

4.4.2. Definir detalles generales para actores

Esto permite estandarizar el diseño de los objetos, sirve de referencia a la hora de crear nuevos actores, además de facilitar la interpretación de la información respectiva de cada objeto.

- Dimensión: con el propósito de homologar las dimensiones de los objetos virtuales con las dimensiones reales, se han de establecer las unidades de medidas para cuantificar dichas características en el mundo virtual y su equivalente en el real.
- Nombre: los nombres deben ser representativos para cada objeto, será apropiado darles nombres de acuerdo al estilo CapWords, además el primer objeto creado se diferencia de los otros con tu extensión ".000".
- Atributos: son las características y propiedades de los objetos, como son propiedades físicas, materiales, texturas, etc. Se utilizará el estilo camelCase
- Acciones: corresponde a las funcionalidades de cada objeto, como son caminar, mover, frenar chocar, etc. Se utilizará el estilo camelCase.

4.4.3. Detalles específicos

Cada actor debe poseer características específicas que lo identifican inequívocamente, ya sea por su nombre o por la clase (tipo general) a la que pertenecen, definiendo los sonidos, las texturas y/o los colores apropiados de cada actor y/o clase.

Así, por ejemplo considerando los objetos auto.000, bus.001, bus.002, los grupos VEHÍCULOS y HUMANOS, se tiene que:

auto.000 puede pertenecer al mismo tipo que bus.001 y bus.002 VEHICULOS, pero se pueden diferenciar en características específicas como su tamaño.

A su vez, bus.001 y bus.002 visualmente pueden ser idénticos además ambos pertenecen al mismo tipo VEHÍCULOS, pero en su nombre se aprecia que no son el mismo actor.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

4.4.4. Definición de acciones y comportamientos

Las acciones de los actores en este caso, están basadas en la vida real y pueden ser triviales para nosotros, pero deben quedar definidas en el ambiente virtual, debido a que son éstas las que condicionan lo que es capaz de hacer o no un actor.

Algunos ejemplos de acciones son caminar, girar, parar, acelerar, frenar, mirar, etc.

Los comportamientos son también considerados como acciones, las que se obtendrán de acuerdo a una situación preparada a priori, por ejemplo puede ser considerado como comportamiento el detener el movimiento de un peatón al presentarse un semáforo en rojo.

Con el fin de simplificar la lógica de acciones, se tiene que un automóvil no puede caminar mientras que un peatón si lo hace, pero ambos se mueven, con lo cual se puede generalizar el desplazamiento de un vehículo (conducción) y el de un peatón (caminar), en una sola acción con el mismo nombre "mover", así como se señala en la metodología[3] basta con programar un objeto, luego duplicarlo y/o unirlos, con la finalidad de utilizar la misma lógica de un objeto en otro para sus acciones y/o comportamiento.

[3] *Introducción práctica a la realidad virtual, Cap. 6 "Enfoques de desarrollo" pág. 115*

Universidad del Bío-Bío. Red de Bibliotecas - Chile

5. METODOLOGÍA DE DESARROLLO Y DISEÑO

La metodología implementada, está basada principalmente en el enfoque propuesto por el profesor Juan Carlos Parra en su libro introductorio a la realidad virtual[4] para el desarrollo de objetos, y además del enfoque propuesto por el LEU, en su página web[5] para el desarrollo e implementación del escenario.

El enfoque establece un análisis general del escenario, además de fijar los objetivos prácticos que tendrá el escenario o ambiente virtual, este análisis se lleva a cabo con una estructura top-down, es decir se percibe el escenario como un objeto único, compuesto de otros objetos más específicos, donde luego se recogen dichos objetos y se detallan aun más.

Después de detallar los objetos se construyen los componentes para una posterior integración en cada uno de los actores, hasta llegar a la construcción total del escenario cumpliendo una secuencia bottom up, empezando desde componentes y llegando al escenario final, el cual cumple los objetivos planteados al inicio del proyecto en desarrollo.

Gráficamente las etapas individuales de diseño y desarrollo para la construcción del escenario virtual de simulación de tránsito vehicular y peatonal son:

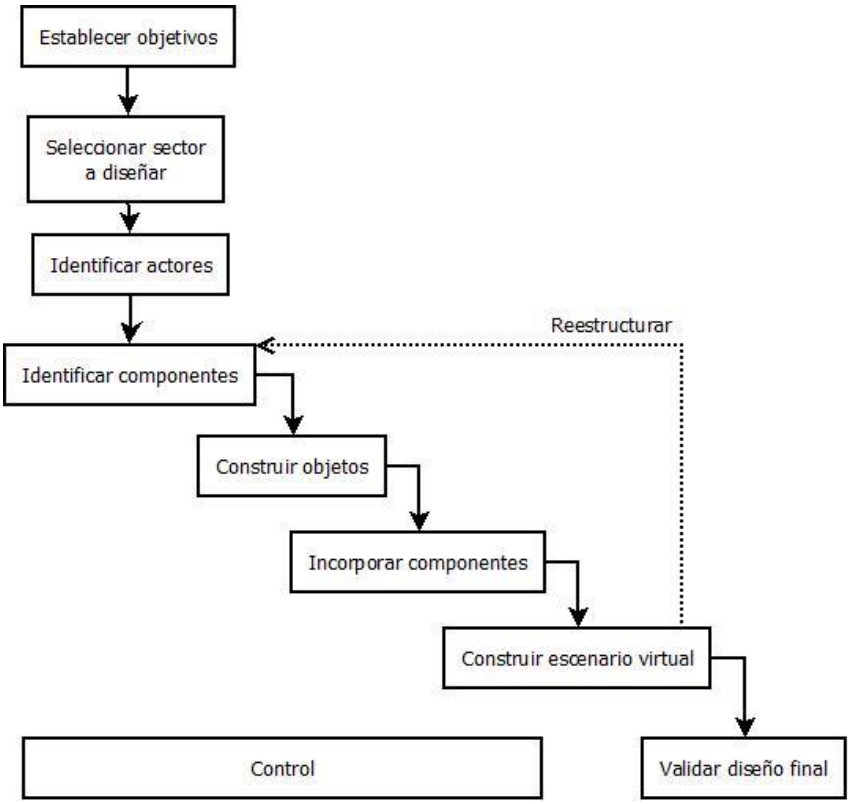


Figura metodología de diseño y desarrollo

[4] Introducción a la realidad virtual 2da edición pág. 116
[5] http://leu.servicios.ubiobio.cl/alfawiki/index.php/Metodología

Universidad del Bío-Bío. Red de Bibliotecas - Chile

En cada etapa antes presentada, es de mucha importancia ir controlando y validando los modelos, así como también las acciones y componentes desarrollados e implementados.

En caso de encontrar errores o problemas en los actores ya identificados, se realiza una reestructuración de ya sea modelos o componentes de objetos diseñados. Dichos problemas es importante luego de ser detectados, solucionarlos a medida que aparecen, debido a que puede darse el caso en que existen objetos que heredan comportamientos, acciones, atributos, propiedades o similares, y al hacer esto, también heredan dichos problemas.

5.1. Establecer objetivos.

Se debe de detectar el o los propósitos por los cuales se ha de diseñar un ambiente virtual, es de importancia debido a que de acuerdo a la finalidad del ambiente virtual o escena a diseñar, dependerán las acciones y grado de detalle de los objetos que interactúen en dicha escena.

Así por ejemplo no es relevante diseñar arboles con gran detalle si la escena los presenta con propósito visual, pero por el contrario, si el diseño está enfocado a un proyecto en el cual se diferencian tipos de arboles, tipo de corteza, hojas, altura, comportamiento durante las estaciones climáticas, etc., dicho proyecto deberá especificar los factores adecuados para alcanzar sus propósitos.

5.2. Seleccionar un sector a diseñar

Para poder diseñar y comenzar la construcción de los actores que interactúan en la escena virtual, se debe identificar la escena a simular, para así seleccionar los objetos necesarios a construir.

Con el propósito de enfocar la construcción de los objetos necesarios, por ejemplo no diseñar un aeropuerto si se fija como objetivo un sector en donde no lo hay.

5.3. Identificar actores

Utilizar un esquema conceptual en donde agrupar los objetos que interactúan en el sector, mediante un esquema conceptual por criterios comunes, en este caso se encuentran categorías para los vehículos, humanos, utilitarios y objetos accesorios.

Con el propósito de evitar diseñar objetos con acciones comunes dos o más veces, podrían agruparse los objetos de acuerdo a otros criterios, como son los objetos móviles y los objetos inmóviles, dado que es una característica relevante y común para ciertos actores, sólo que esta caracterización es muy general para la cantidad de objetos a diseñar.

5.4. Identificar los componentes.

Cada actor se debe descomponer de acuerdo a características significativas, como en este caso los materiales texturas movimientos y sonido si es que los tuviese, para posteriormente permitir su construcción, de una forma cómoda y estructurada.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

5.5. Construir cada uno de los objetos.

Comienza el diseño de cada objeto y su construcción estructural (modelado).
El diseño de los objetos puede ser realizado utilizando figuras geométricas, uniendo vértices y aristas, etc.

5.6. Incorporar componentes a cada objeto.

Luego de crear cada uno de los objetos individualmente, se incorpora a cada uno sus comportamientos y acciones adecuadas esperadas. Es recomendable seguir un orden lógico dando prioridades a algún componente que es contenido en otro, siguiendo la estructura bottom-top.

5.7. Construir el escenario virtual a partir de los objetos.

Finalmente, se construye el escenario virtual utilizando los objetos diseñados forma individual, respetando y buscando satisfacer los objetivos establecidos al inicio del proyecto, en caso de detectar algún problema o aparecer nuevos requerimientos que alteren el diño de algún actor puede reestructurarse volviendo a etapas anteriores.

5.8. Validar diseño final.

Etapa final en donde se busca confirmar el cumplimiento de los objetivos planteados en las etapas iniciales del diseño, mediante pruebas de control del escenario completo con sus actores funcionando en conjunto.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

6. SISTEMA A MODELAR

El diseño de escenarios se realiza en el software de diseño 3D Blender. Este software permite crear escenas 3D simular objetos reales en realidad virtual, además posee motor de juegos 3D integrado (Game Engine), el cual es utilizado para simular el comportamiento de dichos objetos.

6.1. Aspectos generales al diseñar el escenario virtual[6]

Los objetos se diseñan de forma individual en un archivo Blender, cada uno de éstos, posee sus atributos individuales de forma general y a medida que se utilizan en cada escenario se especifica de acuerdo a cada situación esperada.

En relación a las acciones, al utilizar el editor lógico de Blender las acciones se especifican mediante los sensores, el controlador y el actuador.

Sensores: funcionan como detectores de situaciones, acciones, propiedades, materiales entre otros entregando un valor booleano TRUE o FALSE.

Controladores: conectan los sensores con los actuadores principalmente mediante una o más condiciones (excepción el controlador tipo Python).

Actuador: es la acción que se espera al cumplir con las condiciones.

[6] Metodología enfocada en Blender.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

6.1.1. Mundo

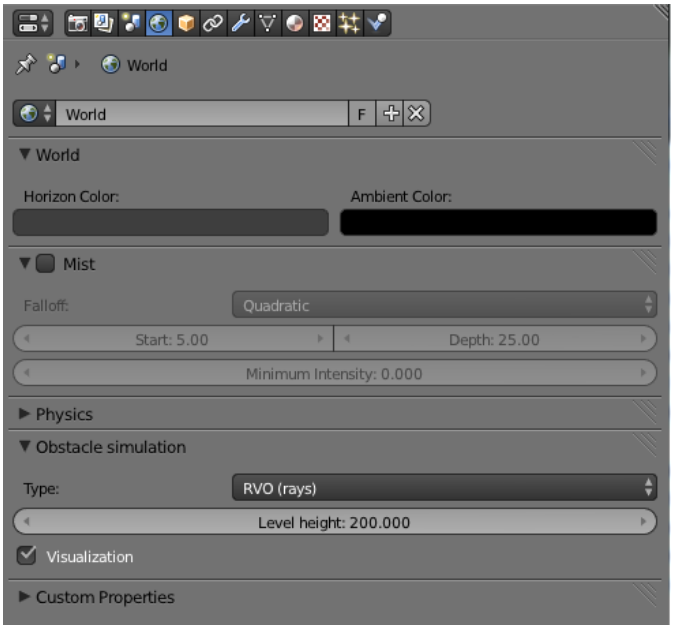
Blender permite a objetos detectar y ser detectado por otros. Uno de los usos apropiados puede ser como función para evitar choques entre dichos objetos.

Activar la simulación de obstáculos y visualización:

Consola

bpy.context.scene.game_settings.obstacle_simulation = 'RVO_RAYS'

bpy.context.scene.game_settings.show_obstacle_simulation = True



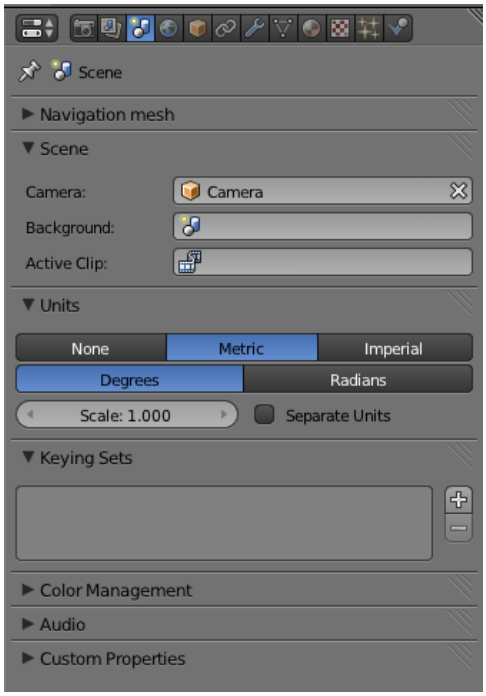
En un principio esta función, es de gran utilidad, pero a medida que se complejizan las acciones de los objetos, dichas acciones pueden generar conflictos con esta función, descontrolando al objeto haciendo su comportamiento impredecible.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

6.1.2. Escena

En la escena se define las unidad de medida para el ambiente virtual a diseñar, en este caso se utiliza la unidad de medida del sistema métrico decimal, donde una unidad de Blender equivale a un metro.

Consola
bpy.context.scene.system = 'METRIC'



Universidad del Bío-Bío. Red de Bibliotecas - Chile

6.1.3. Objetos móviles

Actores que interactúan en el ambiente virtual, principalmente poseen acciones lógicas que les permiten desplazarse por el escenario.

6.1.3.1. Observaciones generales objetos móviles.

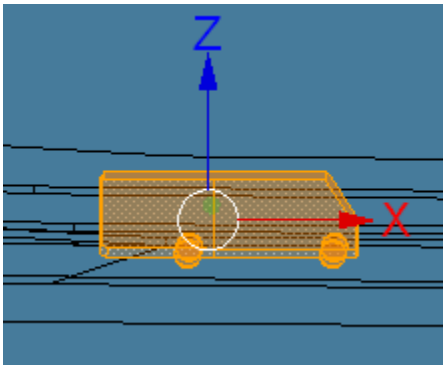


Figura orientación objetos

Es de mucha relevancia el orientar bien un objeto mientras se diseña, ya sea al establecer la posición de las ruedas(que parece obvia eje Z) , y la posición frontal del objeto en el eje +X, la que es muy importante (por lo menos tener clara al momento de diseñar).

Todo esto debido al posterior uso de funciones propias de Blender, como es el actuador steering, donde es fundamental conocer la orientación del objeto utilizando el facing, para guiar al objeto.

Establecer la física del objeto para que pueda interactuar con el mundo, dejándolo como actor dinámico y que permita colisiones.

Consola

bpy.context.object.game.physics_type = 'DYNAMIC'

bpy.context.object.game.use_collision_bounds = True

bpy.context.object.game.collision_bounds_type = 'CONVEX_HULL'

bpy.context.object.game.collision_margin = 0.002

bpy.context.object.game.use_obstacle_create = True

bpy.context.object.game.obstacle_radius = 2

Universidad del Bío-Bío. Red de Bibliotecas - Chile

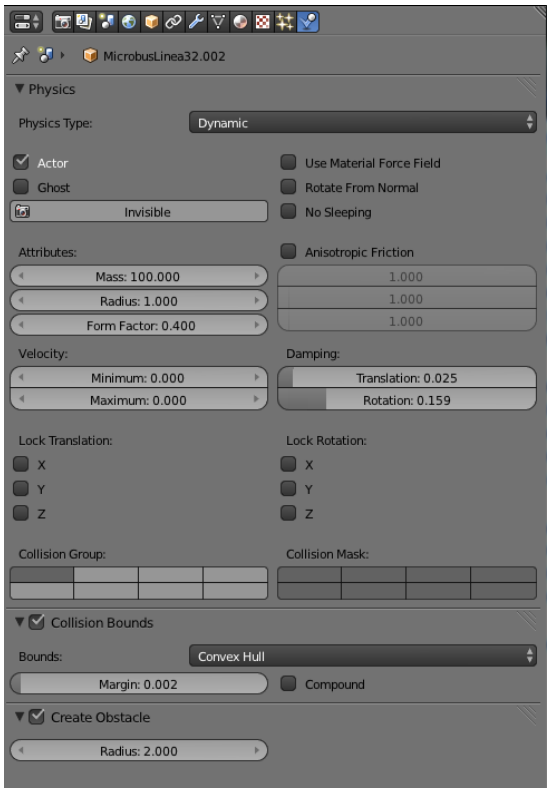


Figura atributos para objetos dinámicos

6.1.3.2. Acciones con objetos móviles.

Como se dijo anteriormente las acciones básicamente son las actividades que puede desarrollar el actor, en Blender usando la Game Engine, y su editor lógico para poder integrar comportamientos a los actores se fijan sensores, los que cumpliendo ciertas condiciones (previamente establecidas) descritas en los controladores activan o desactivan los actuadores, permitiendo (o no) que el objeto realice la acción.

Figura lógica de acciones

El sensor puede no necesariamente enviar una señal positiva (true), también puede activar el controlador al no estar activo invirtiéndolo (estado false), todo dependiendo de cómo se haya definido.

El Controlador por su parte, puede recibir señales de propiedades definidas en el objeto, también puede estar condicionado por un script escrito en lenguaje Python, permitiendo o no al actuador realizar su acción (el actuador puede estar incluido en el controlador por medio del script).

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Estos sensores son usados para simulan el comportamiento de la visión y detección de objetos determinados utilizando las funcionalidades de Blender, principalmente los tipos Ray, Radar y Collision.

Se diferencian dos grupos de sensores principalmente los de detención (parar) y los de frenar (cambia la velocidad actual).

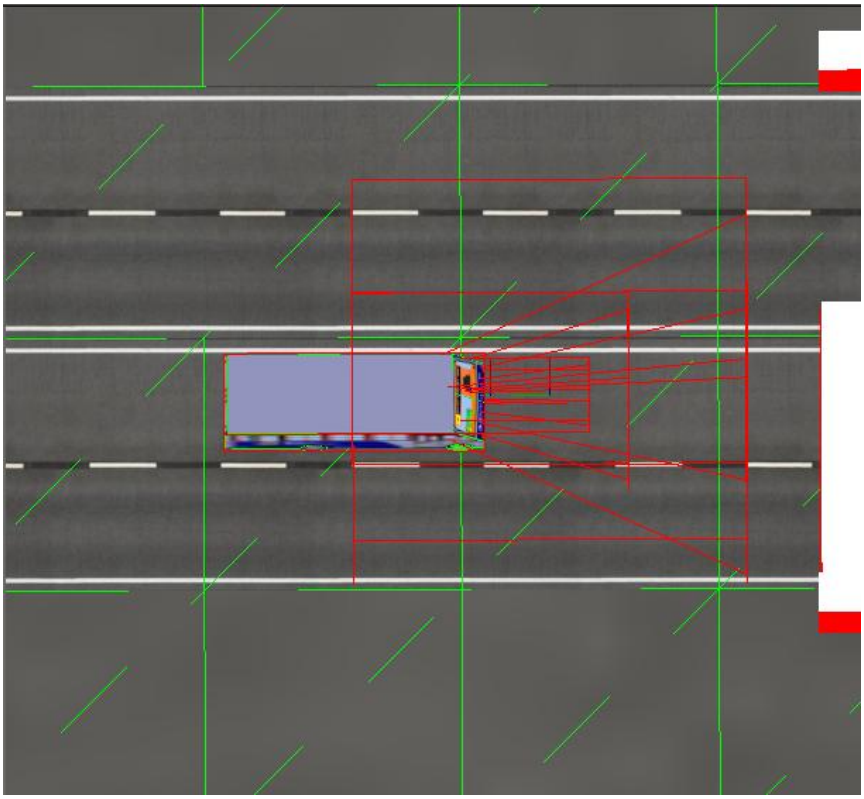


Figura rango de detención y rango de frenado.

En la figura se aprecian los sensores de tipo rayo y tipo radar, utilizados para poder detectar cierto tipo de objetos, como las personas, otros vehículos y semáforos. Activando los actuadores respectivos a cada sensor.

Los rangos se pueden modificar para simular ciertas características, por ejemplo para poder simular a un conductor precavido, puede aumentar el valor de la variable que permite la detección de vehículos y otros objetos.

Por el contrario si se necesita simular conductores y/o peatones imprudentes disminuye los valores necesarios.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Blender posee una gran variedad de sensores, los sensores utilizados principalmente son los siguientes:

Tipo Ray (rayo):

Tiene propiedades de rango para la distancia con la cual se activará (o desactivará), el material a detectar, el eje por donde se proyectara el rayo y una propiedad que permite atravesar objetos que estén dentro del rango.

Figura Sensor tipo ray.

El sensor de tipo rayo se representa de color verde claro, lo que permite conocer el rango de una manera visual.

En este caso se proyecta por la parte frontal del objeto, simulando la distancia de frenado para el vehículo.

Por ejemplo: para la detección de semáforos en color rojo (parar).

Se proyecta un rayo que al detectar un objeto de material "rojo" activa la detención.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Tipo radar

Tiene propiedades de distancia y ángulo con la cual se activará (o desactivará) el sensor. Además, posee la propiedad a detectar, y el eje por donde se proyectará el radar.

Este tipo de sensor se puede utilizar para simular el rango de visión humana mientras se conduce (vehículo), o mientras se camina (es peatón).

Por ejemplo:

Se utiliza para la detección de vehículos en cruces peligrosos donde dentro del rango y ángulo de detección, el vehículo frena al detectar otro vehículo.

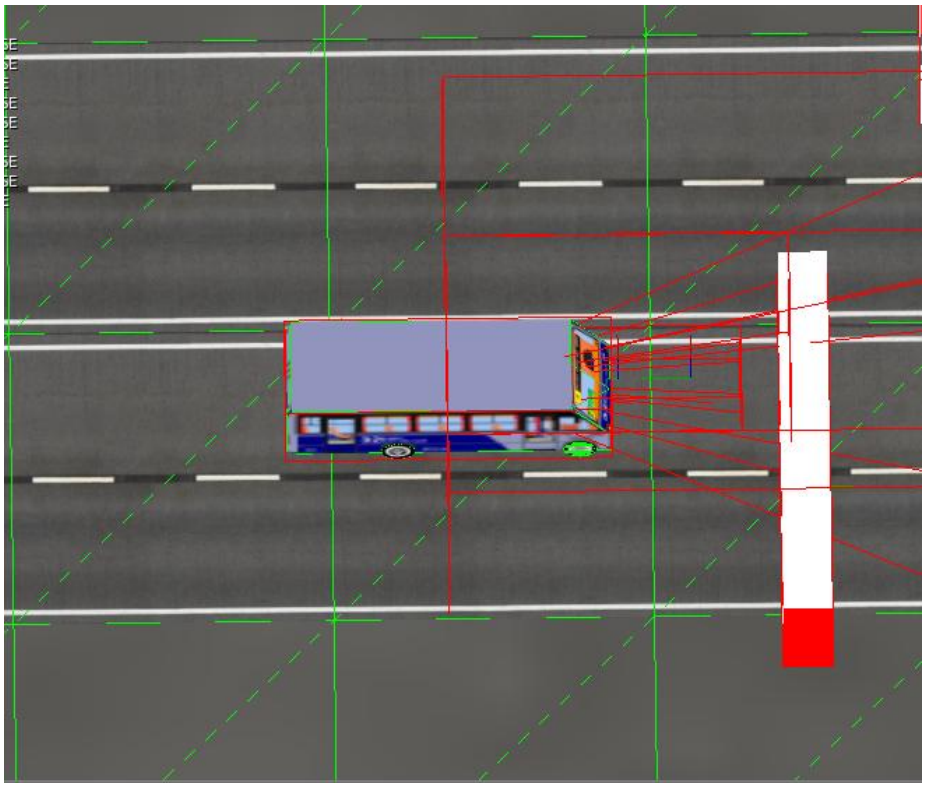


Figura Sensor tipo radar.

En la imagen se aprecia la forma de cono, que proyecta este sensor y se visualiza con un color rojo.

En este caso se utiliza para simular el rango de visión que posee un conductor.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Tipo Collision (colisión).

Este sensor detecta el choque con otro objeto de un determinado material o propiedad.

Por ejemplo:

Se utiliza para detectar choques entre vehículos, al estar activo este sensor, el objeto deja de moverse.

Figura Sensor tipo collision.

En la figura se aprecia cómo funciona el sensor rodeando el objeto, detectando el ambiente por donde circula.

Tipo Always (siempre)

Es usado para poder iniciar las acciones del objeto.

Observación.

Activar "use_pulse_true_level", para poder refrescar el sensor constantemente.

Consola

bpy.context.object.use_pulse_true_level = True

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7. DESARROLLO DISEÑO

Desarrollo del escenario virtual planteado como parte del proyecto, utilizando la metodología anteriormente expuesta.

7.1. Establecer objetivos.

En este caso como es un ambiente orientado al tránsito de vehículos y personas, el objetivo es poder simular un ambiente que sea indiferenciado a la realidad física en donde circulen vehículos y personas.

7.2. Seleccionar un sector a diseñar

Se utilizó como sector objetivo, la intersección de las avenidas Collao, Irarrázaval y Los Carrera.

Figura guía del escenario a diseñar[7]

[7] *https://maps.google.com/maps?ll=-36.814915,-73.029594&spn=0.003638,0.003846&t=h&z=18*

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.3. Identificar actores

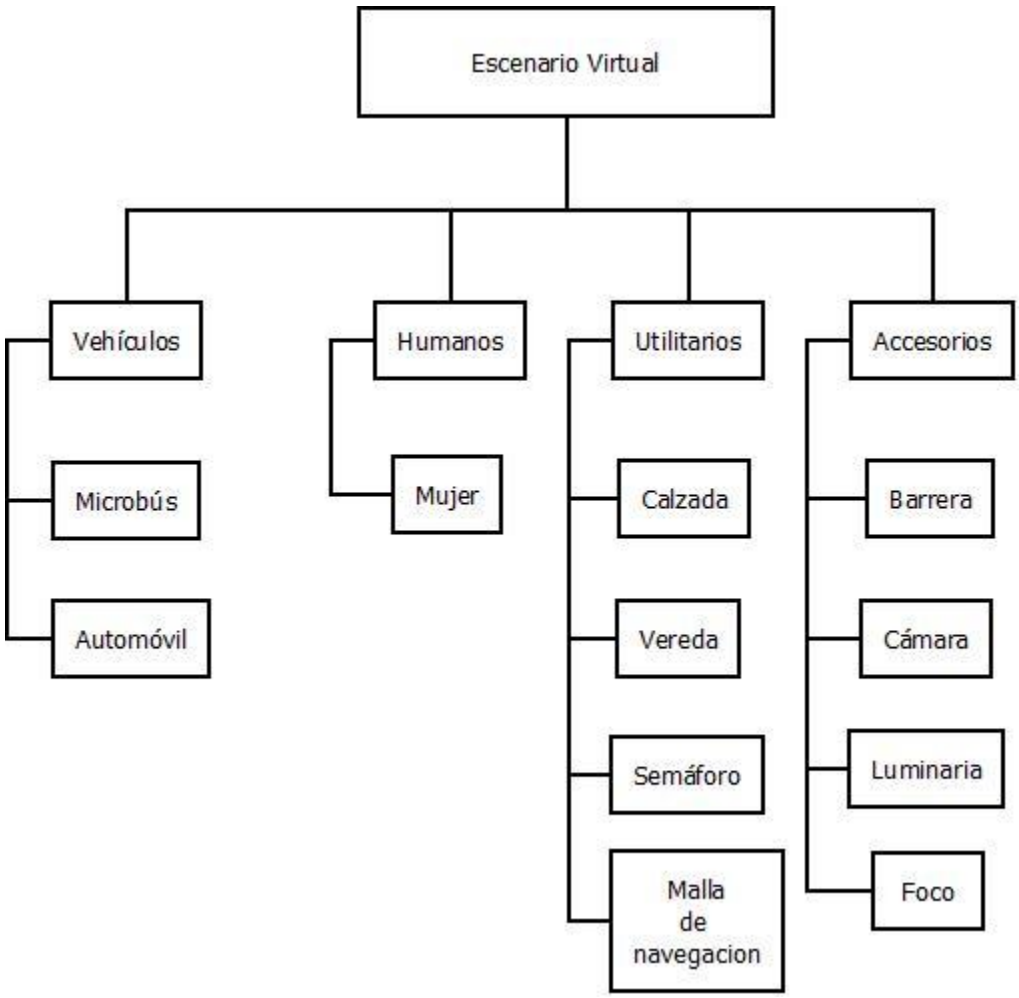
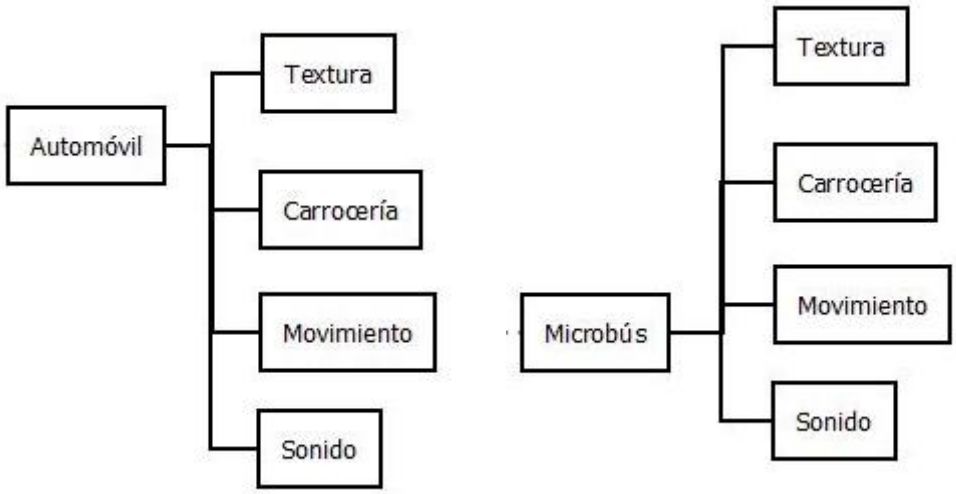


Figura esquema conceptual de objetos

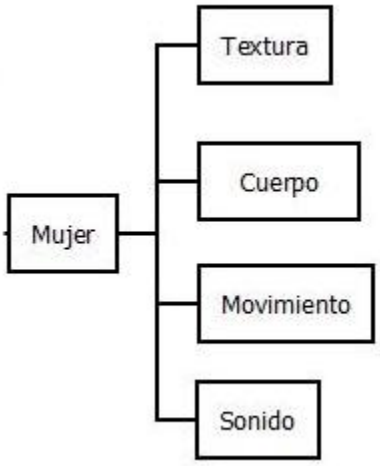
Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.4. Identificar los componentes.

Vehículos:

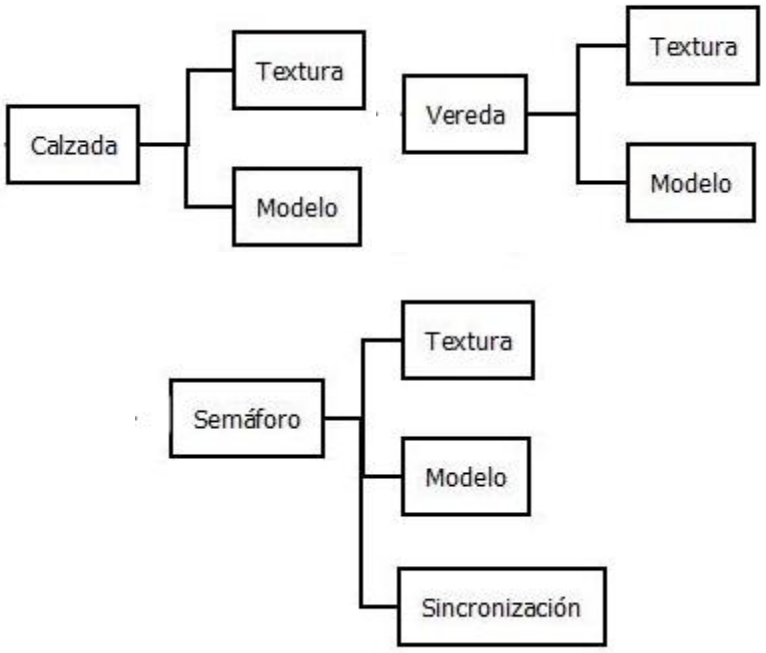


Humanos:

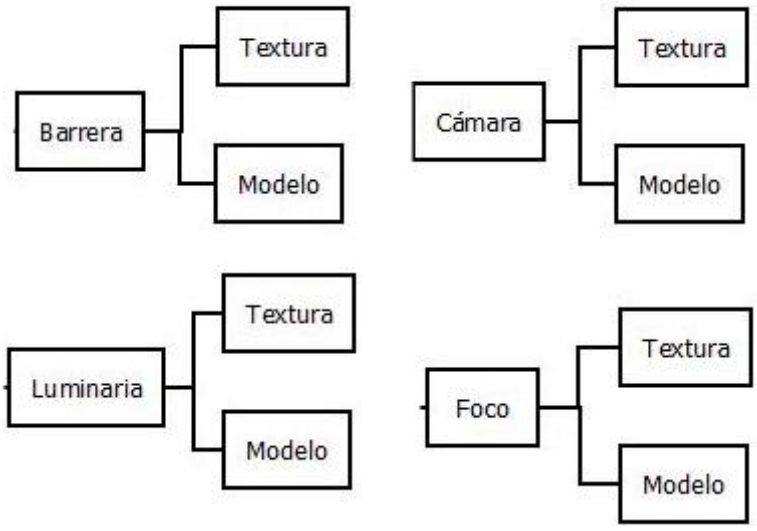


Universidad del Bío-Bío. Red de Bibliotecas - Chile

Utilitarios:



Accesorios:



Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.5. Construir cada uno de los objetos.

Para la construcción de los modelos implementados en el proyecto, se utilizan diferentes tipos de herramientas entregadas por el software 3D en este caso Blender.

Aquí se detalla como modelar de una manera general, un objeto con esta herramienta.

Figuras Primitivas

Las figuras primitivas, son la base para el diseño de un modelo 3D, la cual se irá deformando de su estado inicial para obtener la figura deseada, ya sea utilizando un simple cubo, esfera, plano o similar.

Modo	Object Mode, Edit Mode
Panel	3D view
Comando	Shift + A -> Seleccionar Mesh

Tipos de vistas

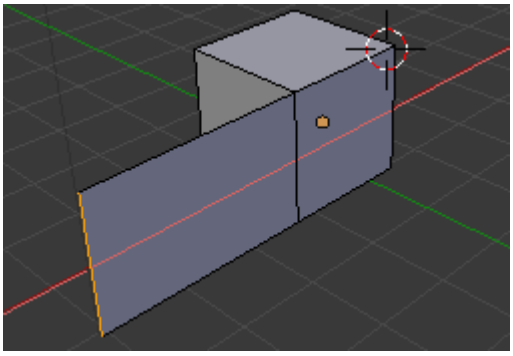
Para el modelado es muy importante la perspectiva con la que se está trabajando, es por esto, que se cuenta con diferentes tipos de vistas predeterminadas, como por ejemplo una vista desde sólo el eje X, Y, Z o una vista ortogonal.

Modo	Todos
Panel	3D view
Comando	Teclas numéricas

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Movimiento hacia un eje

El hecho de que se trabaje en un espacio 3D, hace que el modelado hacia un eje como por ejemplo poner una nueva cara a la figura se vuelva bastante complejo, sobre todo si se quiere mantener una simetría con ella, es así que el mismo programa ofrece herramientas para diseñar hacia un eje (X,Y,Z).

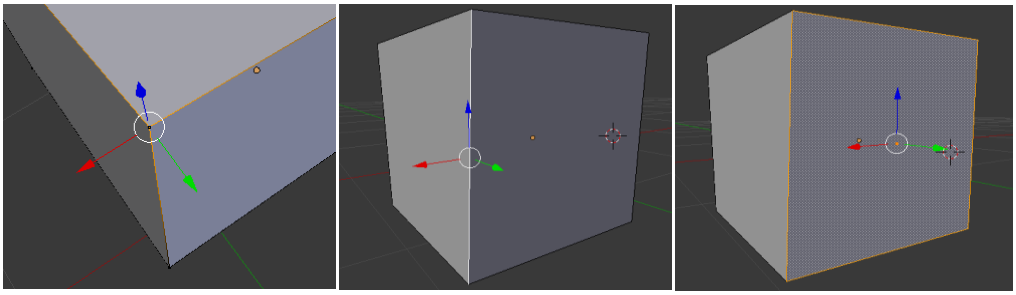


Modo	Edit Mode
Panel	3D view
Comando	Eje X : Tecla ‘x’ Eje Y : Tecla ‘y’ Eje Z: Tecla ‘z’

Selección de aristas, vértices y caras

Para la manipulación de una figura que esté siendo modificada, se puede hacer con tres herramientas básicas, como son:

1. Selección por arista
2. Selección por vértice
3. Selección por cara



Modo	Edit Mode
Panel	3D view
Comando	-

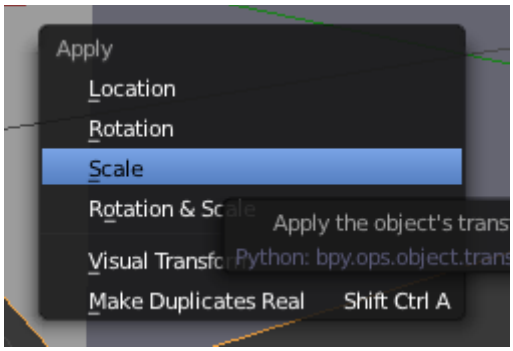
Universidad del Bío-Bío. Red de Bibliotecas - Chile

Ajustar escala de figura y unidad de medida

Uno de los problemas es poder crear objetos a escala, definido como por ejemplo en metros, si el objeto no tiene una escala de (1,1,1) , los valores en metros serán incorrectos.

Ajustar valores a (1,1,1):

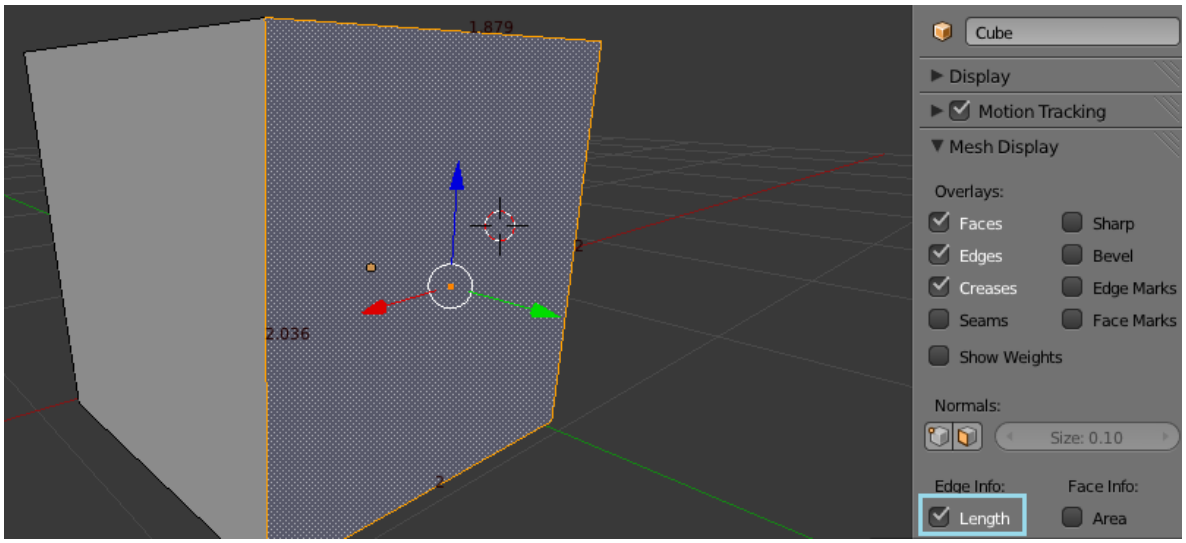
Modo	Object Mode
Panel	3D view
Comando	Control + A



Asignar medidas al modelo:

Modo	Object Mode
Panel	Transform
Comando	-

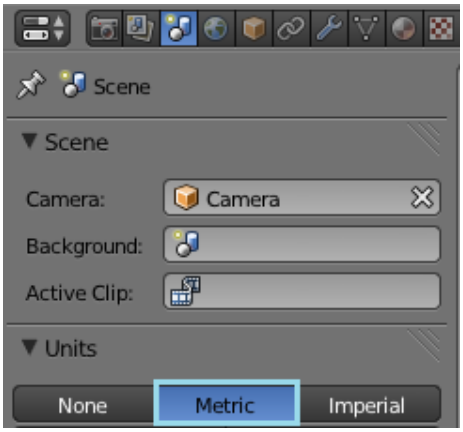
Seleccionar Lenght.



Universidad del Bío-Bío. Red de Bibliotecas - Chile

Asignar unidad de medida:

Modo	Object Mode
Panel	Properties -> Scene
Comando	-



Ajustar escala (Tamaño) libremente:

Modo	Edit Mode
Panel	3D view
Comando	Tecla ‘s’

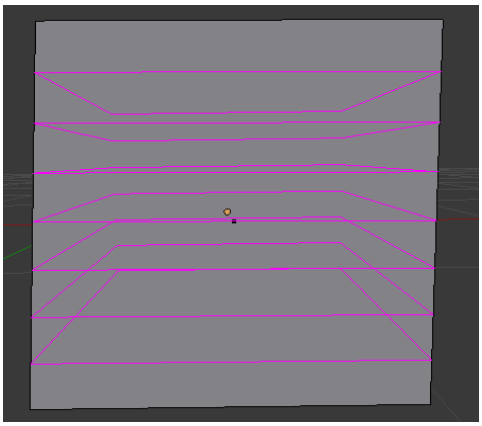
Universidad del Bío-Bío. Red de Bibliotecas - Chile

Sub-división de cara

Divide una figura en N cantidad.

Modo	Edit Mode
Panel	3D view
Comando	Control + R -> Acercar puntero a una esquina de la figura

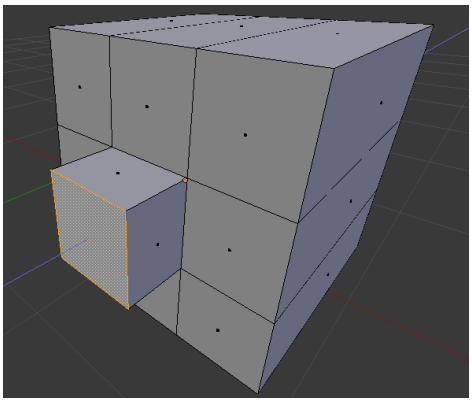
Usar scroll para sub-dividir en más cantidad de líneas.



Mover caras dentro de una figura

Mover la posición de una cara a un nuevo punto de ubicación.

Modo	Edit Mode
Panel	3D view
Comando	Tecla ‘E’ con la cara seleccionada

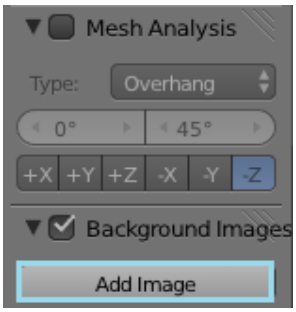


Universidad del Bío-Bío. Red de Bibliotecas - Chile

Asignación de foto de fondo

En el área de trabajo permite asignar una imagen de fondo, útil para modelar algún objeto perteneciente a la foto.

Modo	Object Mode
Panel	Transform
Comando	-



Eliminar un Objeto/Vértice/Arista

Modo	Edit Mode
Panel	3D View
Comando	Tecla ‘x’

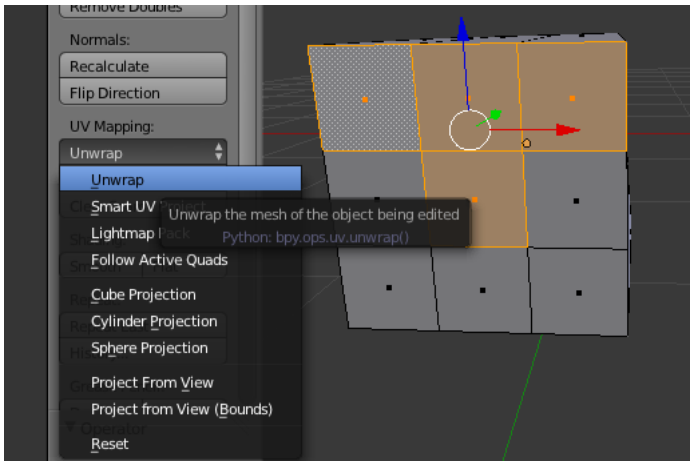
Universidad del Bío-Bío. Red de Bibliotecas - Chile

Asignar textura a un objeto

Blender permite asignar texturas ya sea de tipo jpg, png o usar el panel de colores que viene integrado.

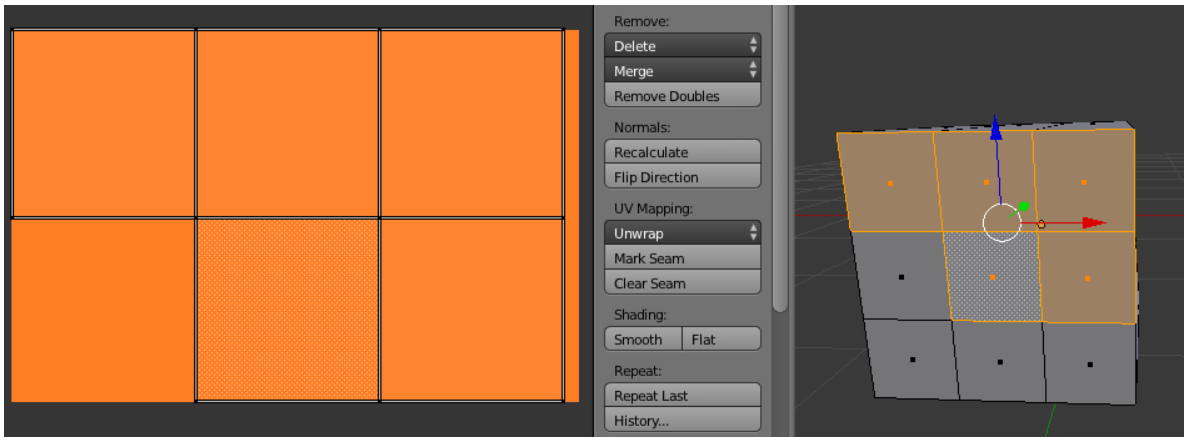
Modo	UV Editing
Panel	Mesh Tools
Comando	-

Seleccionar las caras a las que se desean asignar una textura o pintar, luego cliquear Unwrap.



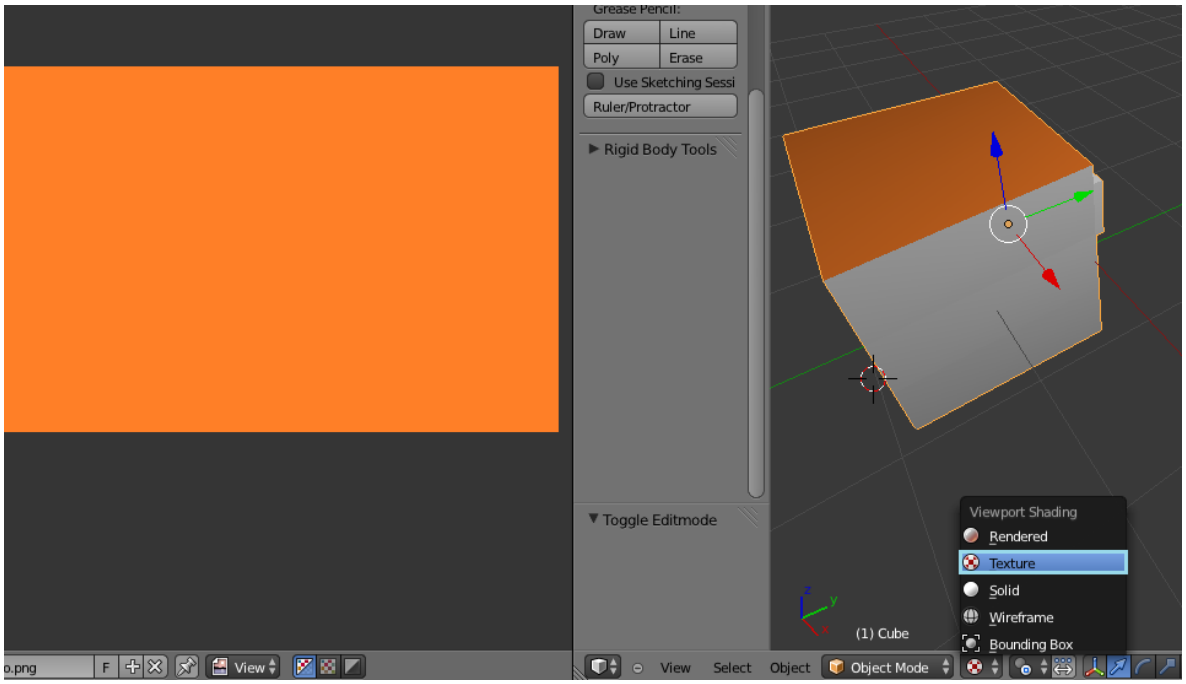
Seleccionar una imagen:

Ajustar en el panel UV Editing la imagen con las caras que fueron seleccionadas.



Universidad del Bío-Bío. Red de Bibliotecas - Chile

Para poder observar las texturas asignadas al modelo 3D, seleccionar en el panel 3D view en la opción Viewport shading 'Texture'



Snap during transform (Imán)

Se utiliza para unir y atraer de una manera efectiva elementos primitivos tales como vértices, líneas y caras de la figura en edición.

Se puede encontrar en la barra de herramientas que entrega el modo 3D VIEW. Este se encuentra ubicado en el menú Editor Type a la izquierda de la interfaz de Blender.

Modo	Object Mode, Edit Mode
Panel	3D view
Comando	Shift + Tab

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Cambiar de Layer (Capa)

Blender ofrece la posibilidad de administrar los objetos que se modelan y asignarlo a una capa determinada.

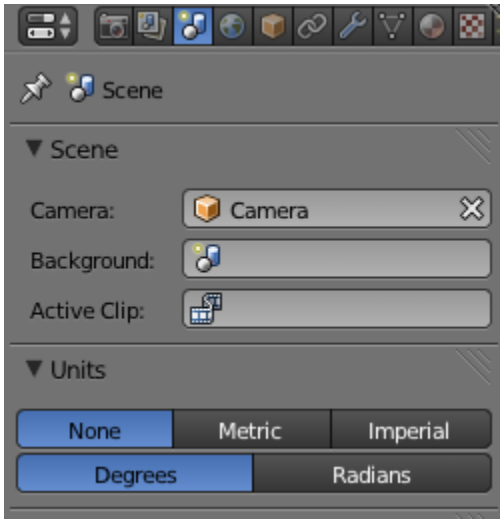
MovetoLayer (Mueve a la Capa)

Modo	Object Mode
Panel	3D view
Comando	Tecla ‘M’

Se utiliza para separar partes de un cuerpo y así poder editar solo la parte deseada sin afectar a las demás.

Una vez ya en el menú de MovetoLayer puedes seleccionar varias capas a la vez con el comando Shift + las Capas a deseadas.

Scene (Escena)



En el menú de propiedades se encuentran múltiples opciones, entre estas Scene, para utilizarla se debe seleccionar el icono , que te entregara herramientas de edición para los elementos de la interfaz, siendo el más importante el sistema de unidades para poder crear un modelo a escala, o sea, modelarlo a medida.

Se podrá modelar con unidades de tipo métrica, imperial, gradual y en radianes.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Desplazamiento

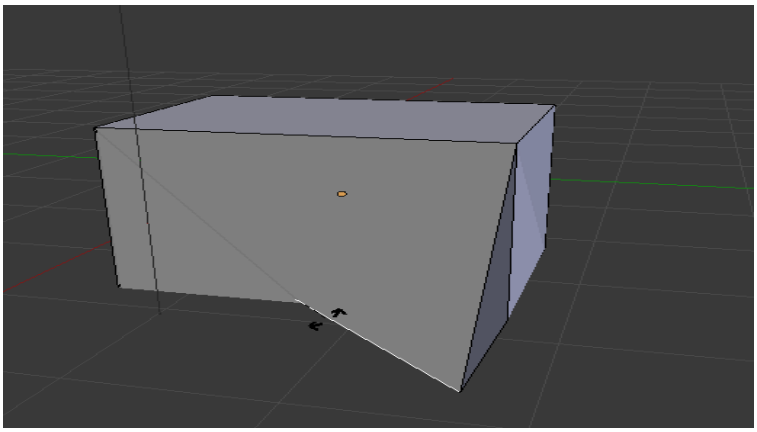
Modo	Object Mode, EditMode
Panel	-
Comando	Seleccionar Elemento + G

Se Puede utilizar tanto en ObjectMode como en EditMode. En el primer, solamente moverá el objeto completo, en cambio, en EditMode moverá el vértice, línea o cara según este seleccionado.

Extrudir

Expande y mueve el elemento primitivo seleccionado involucrando todo lo que esté conectado al mismo.

Modo	EditMode
Panel	Mesh Tools -> Extrude
Comando	Tecla ‘E’

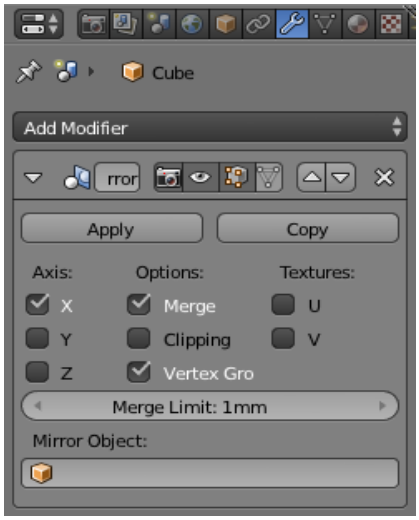


Universidad del Bío-Bío. Red de Bibliotecas - Chile

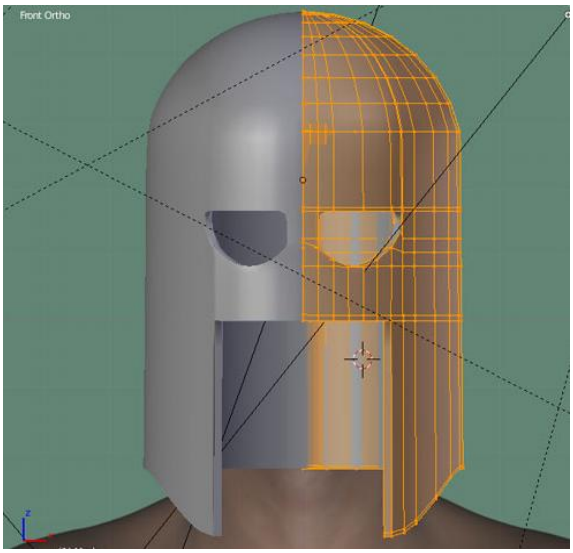
Modificador Mirror (espejo)

Se puede utilizar este modificador para ahorrar trabajo al momento de la creación de un modelo, ya que toda la edición realizada será duplicada al otro lado del eje seleccionado.

Modo	Todos
Panel	OutLiner ->AddModifier
Comando	-



Ejemplo:

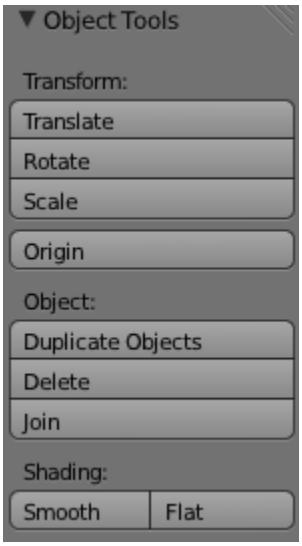


Universidad del Bío-Bío. Red de Bibliotecas - Chile

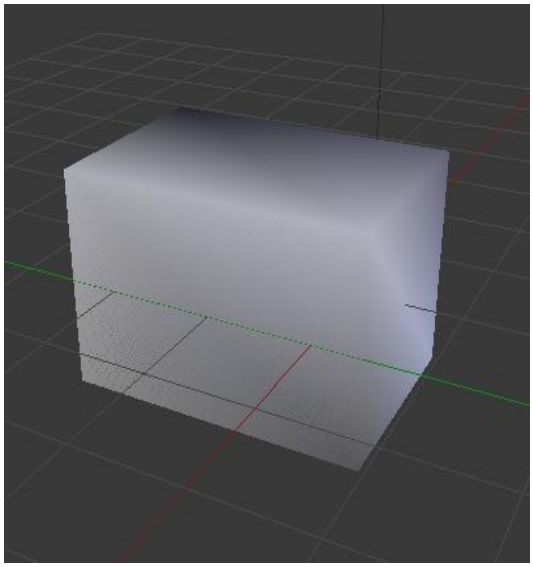
Shading

Esta es la opción de sombreado, para efectos ópticos produce la sensación de suavizado.

Modo	EditMode
Panel	Object Tools -> Shading -> Smooth
Comando	-



Ejemplo:



Universidad del Bío-Bío. Red de Bibliotecas - Chile

Vista Panorámica

Nos proporciona un movimiento libre de vista dentro de los distintos modos que entrega Blender.

Modo	Todos
Panel	-
Comando	Shift + Botón medio del Mouse (Rueda)

Zoom en Detalle

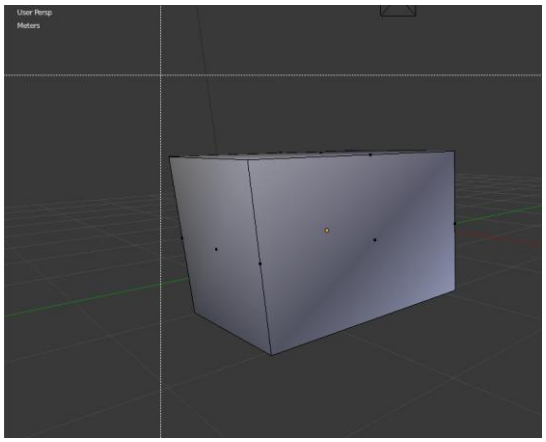
Este comando ayuda a tener un mayor control sobre el zoom aplicado sobre un punto y vista específico.

Modo	Todos
Panel	-
Comando	Ctrl + Botón medio del Mouse (Rueda)

Seleccionar por áreas

Esta opción sirve para seleccionar sólo lo deseado en un área determinada.

Modo	EditeMode
Panel	-
Comando	Tecla ‘B’

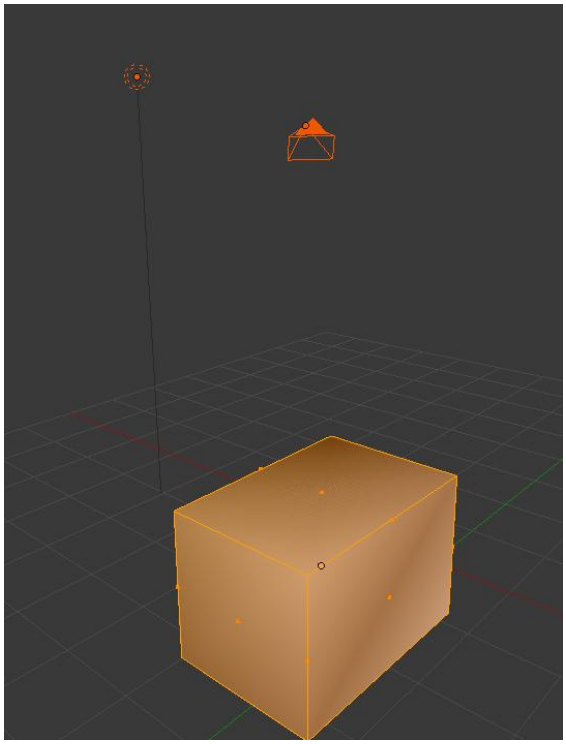


Universidad del Bío-Bío. Red de Bibliotecas - Chile

Seleccionar/Deseleccionar

Este comando ayuda a seleccionar todo lo que se encuentra en la escena.

Modo	ObjectMode, EditeMode
Panel	-
Comando	Tecla ‘A’



Universidad del Bío-Bío. Red de Bibliotecas - Chile

Los principales objetos utilizados en este proyecto se presentan a continuación

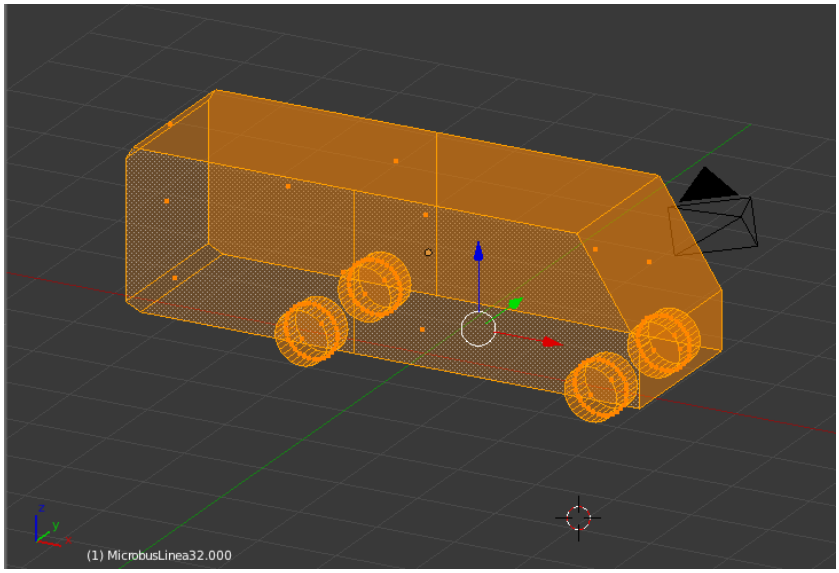


Figura diseño construcción Microbús

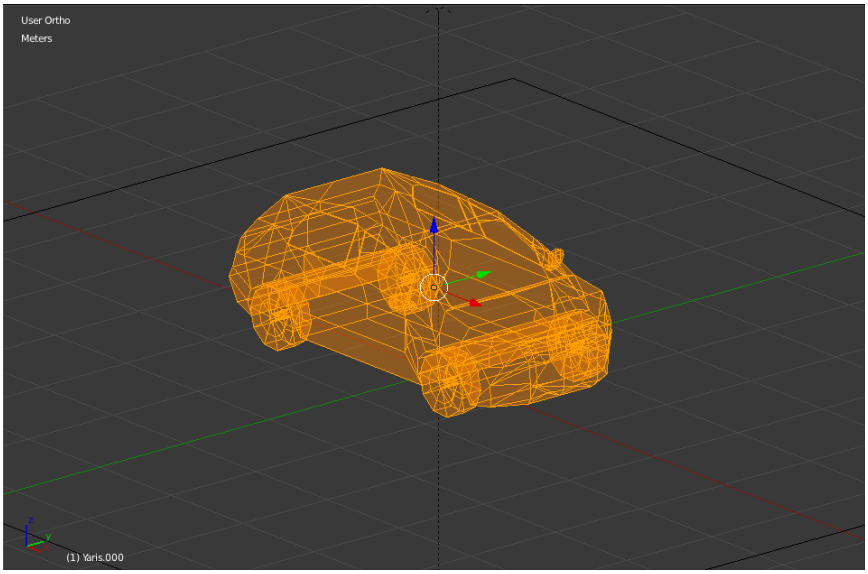


Figura diseño construcción Automóvil

Universidad del Bío-Bío. Red de Bibliotecas - Chile

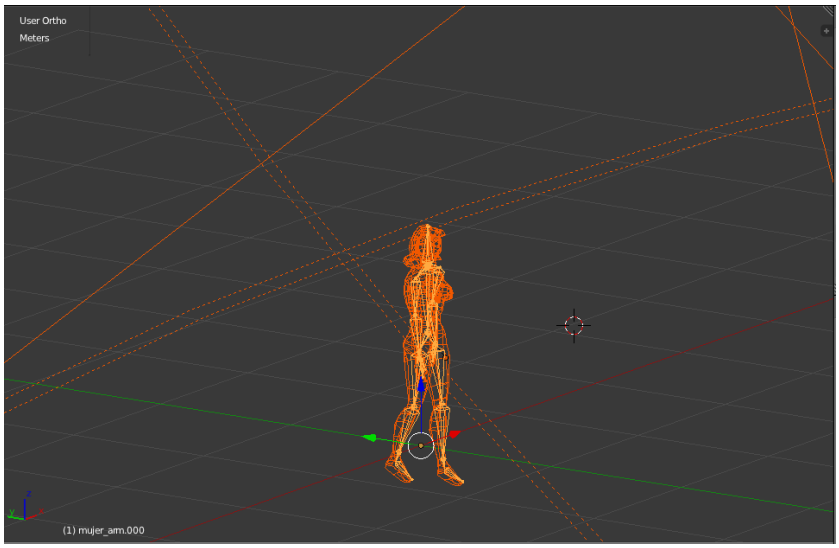


Figura diseño construcción Mujer

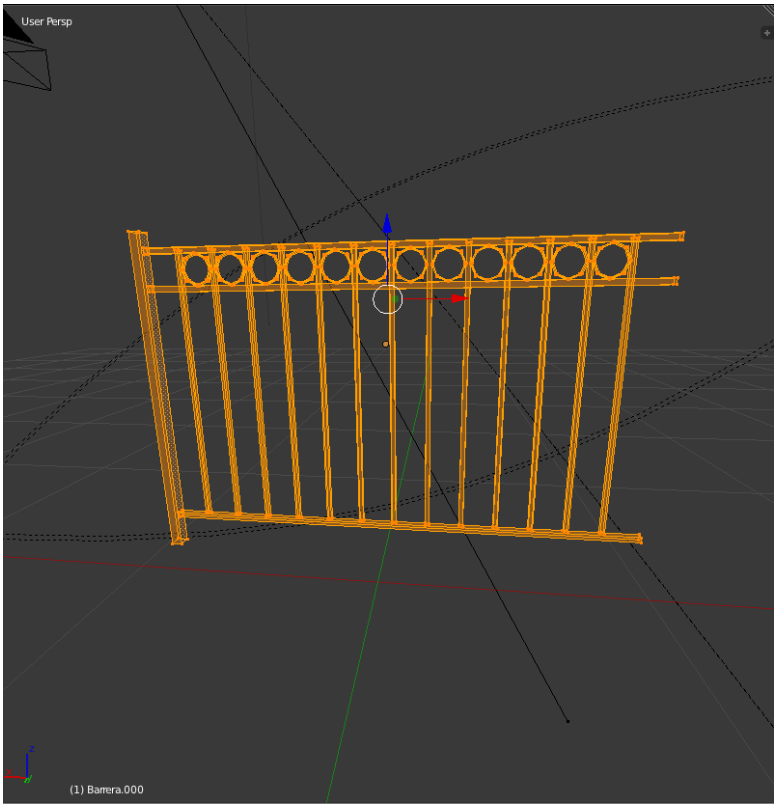


Figura diseño construcción Barrera

Universidad del Bío-Bío. Red de Bibliotecas - Chile

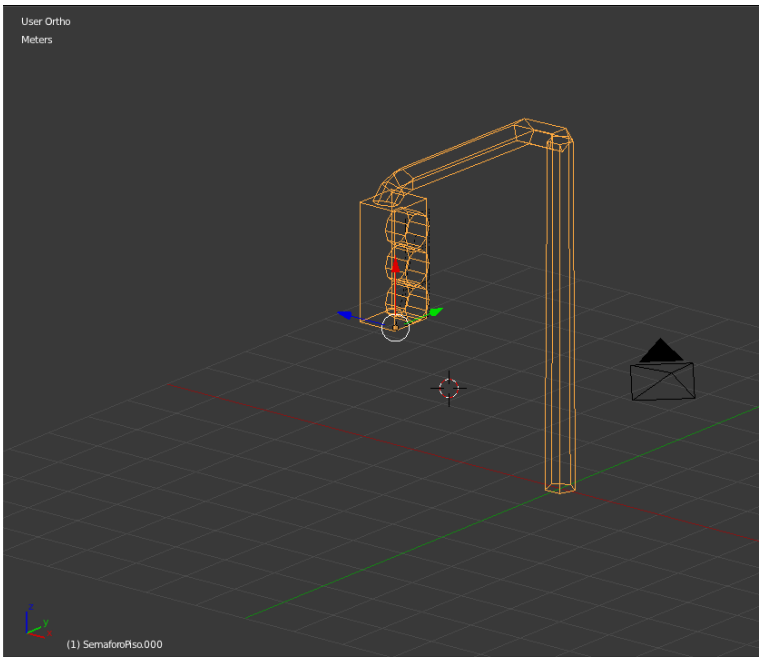


Figura diseño construcción Semáforo

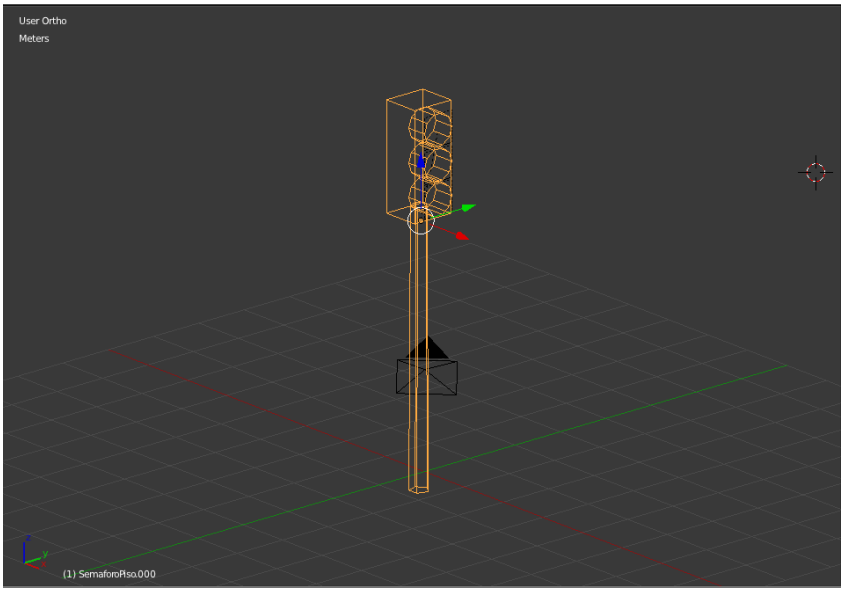


Figura diseño construcción Semáforo piso

Universidad del Bío-Bío. Red de Bibliotecas - Chile

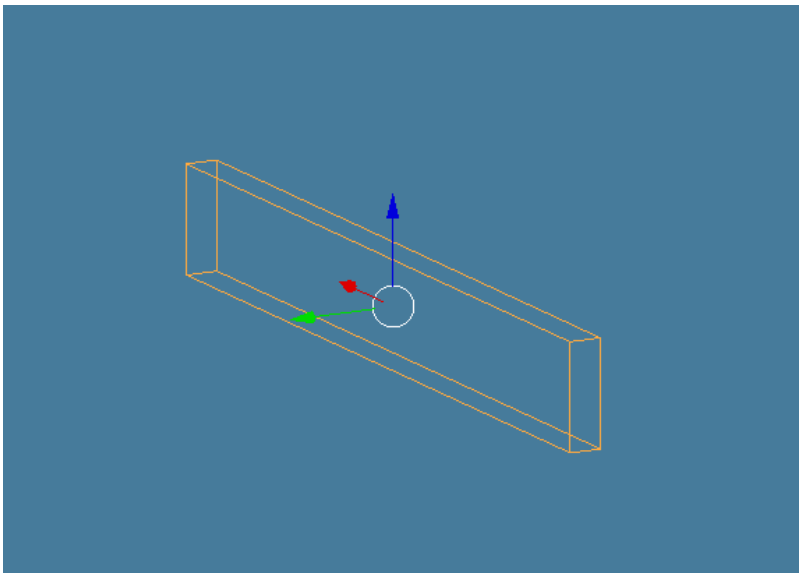


Figura diseño construcción Barrera semáforo

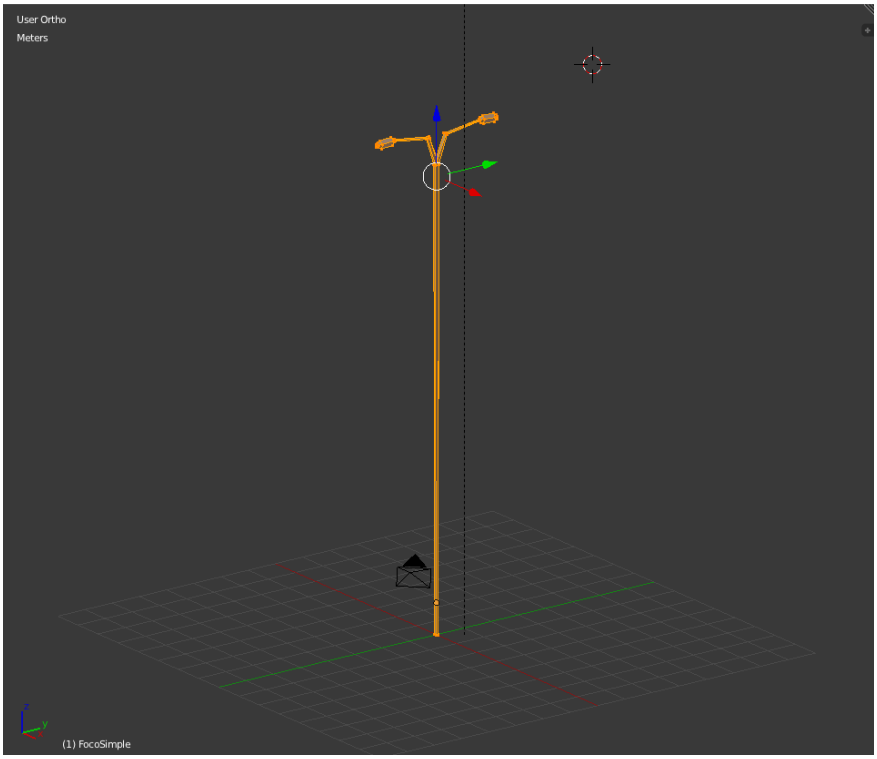


Figura diseño construcción Foco Doble

Universidad del Bío-Bío. Red de Bibliotecas - Chile

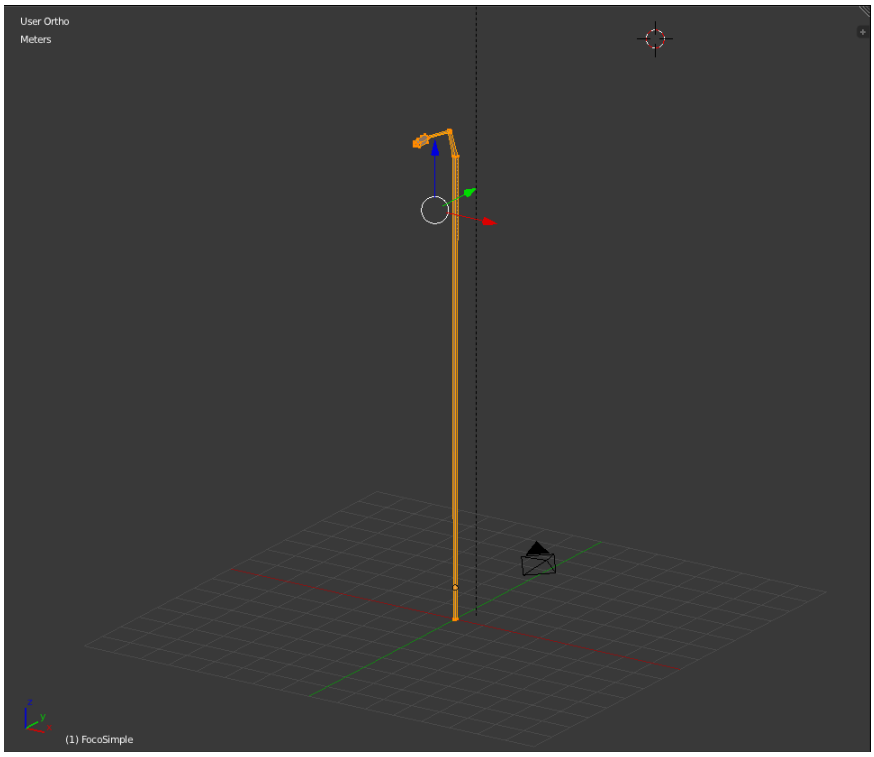


Figura diseño construcción Foco simple

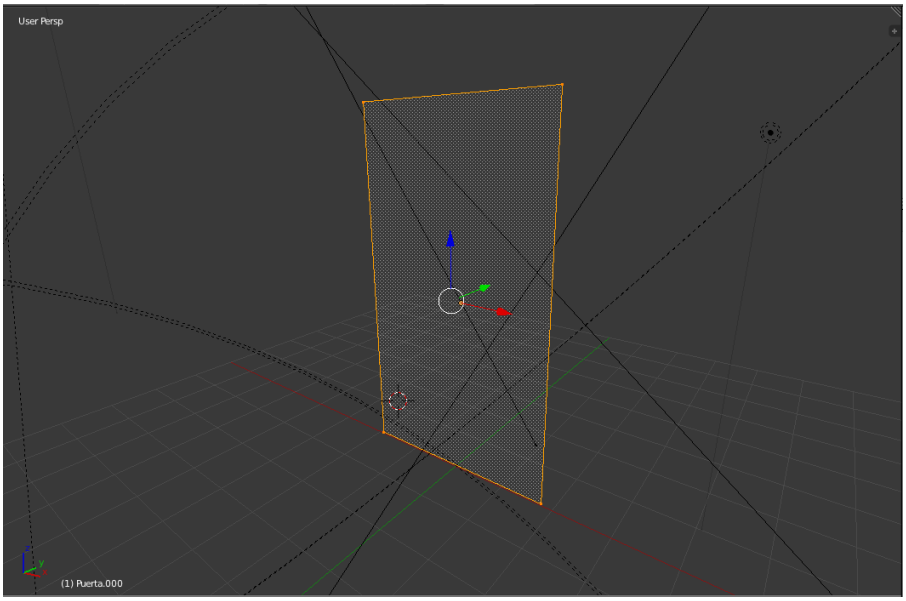


Figura diseño construcción Puerta

Universidad del Bío-Bío. Red de Bibliotecas - Chile

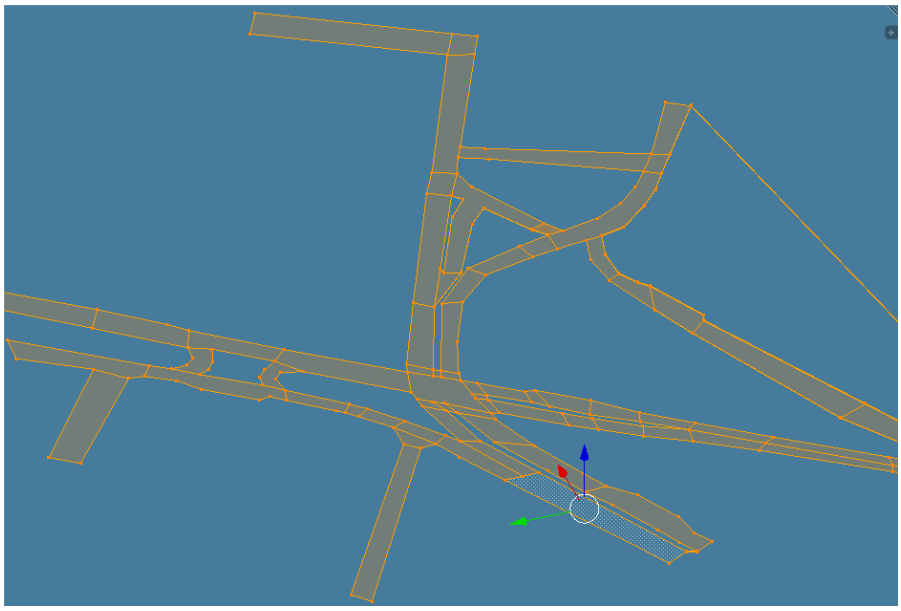


Figura diseño construcción Calzada

Una buena técnica y que fue utilizada al momento de diseñar la calzada, es utilizar la imagen de referencia como fondo y trazar los puntos por donde existe calzada realmente así:

Figura diseño calzada imagen de fondo como referencia

Otra forma es importar los datos desde Open Street Maps[8]

[8] http://leu.servicios.ubiobio.cl/alfawiki/index.php/Metodología

Universidad del Bío-Bío. Red de Bibliotecas - Chile

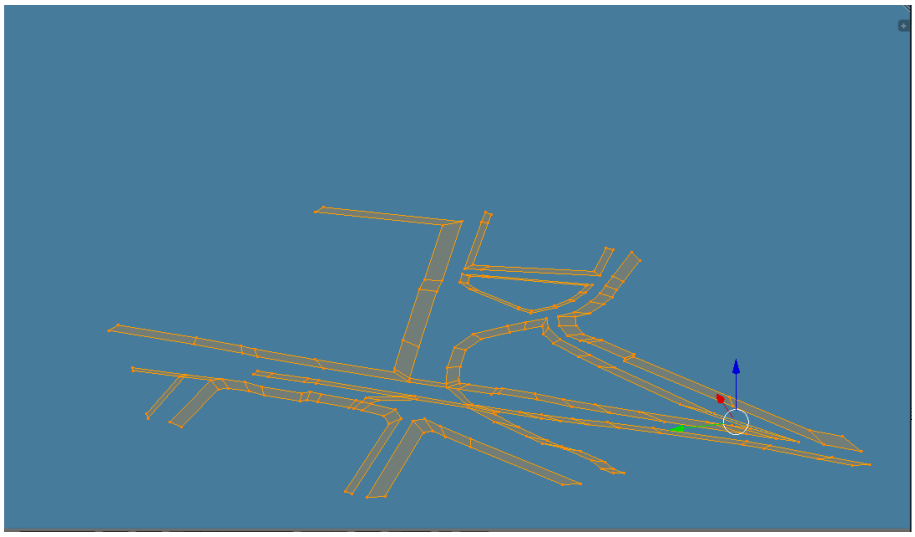


Figura diseño construcción Vereda

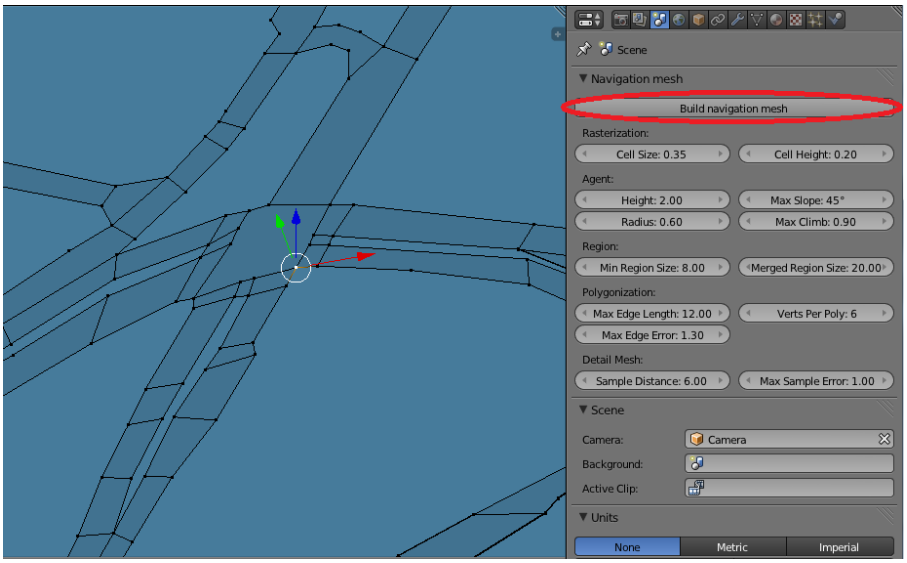


Figura diseño construcción Malla de navegación.

La forma de generar una malla de navegación es seleccionando la superficie por donde se requiere la circulación de objetos en este caso es la calzada para los vehículos y la vereda para los peatones. Luego busca la propiedad escena y se pulsa "build navegation mesh".

En el caso del desarrollo de este proyecto no fue necesario modelar las cámaras y fuentes de iluminación, debido a que ya vienen incluidos en el software.
Principalmente el diseño de los modelos presentados se realiza comenzando con un plano o bien a partir de la figura básica un cubo[9].

[9] Blender 2.68

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.6. Incorporar componentes a cada objeto.

Luego de crear cada uno de los objetos individualmente, se incorpora a cada uno sus comportamientos adecuados esperados.

7.6.1. Componentes de vehículos:

Propiedades para vehículos.

Las propiedades de los vehículos sirven para identificar la acción que está realizando por ello:

- Parar, será positivo cuando el vehículo deba detenerse.
- Choque, positivo luego de que el vehículo tenga algún tipo de colisión detectada.
- Frenar, será positivo cuando el vehículo deba reducir su velocidad.
- La propiedad vehículo se utiliza para dar a conocer a los demás objetos el tipo al que pertenece.

Lógica de acciones para vehículos:

Lógica parar true

Usa sensores para activar la propiedad de "parar", con el fin de detener el vehículo mientras se cumpla una de estas condiciones.

Detecte:

- Material tipo rojo en un objeto, dentro de un rango inferior o igual a 5 ub, y es proyectada por el eje x positivo del objeto.
- Propiedad tipo vehículo en un objeto, a una distancia de 6 ub, es proyectada por el eje x positivo del objeto y con una amplitud de 25º.
- Propiedad tipo persona en un objeto, a una distancia de 7 ub, es proyectada por el eje x positivo del objeto y con una amplitud de 35º.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

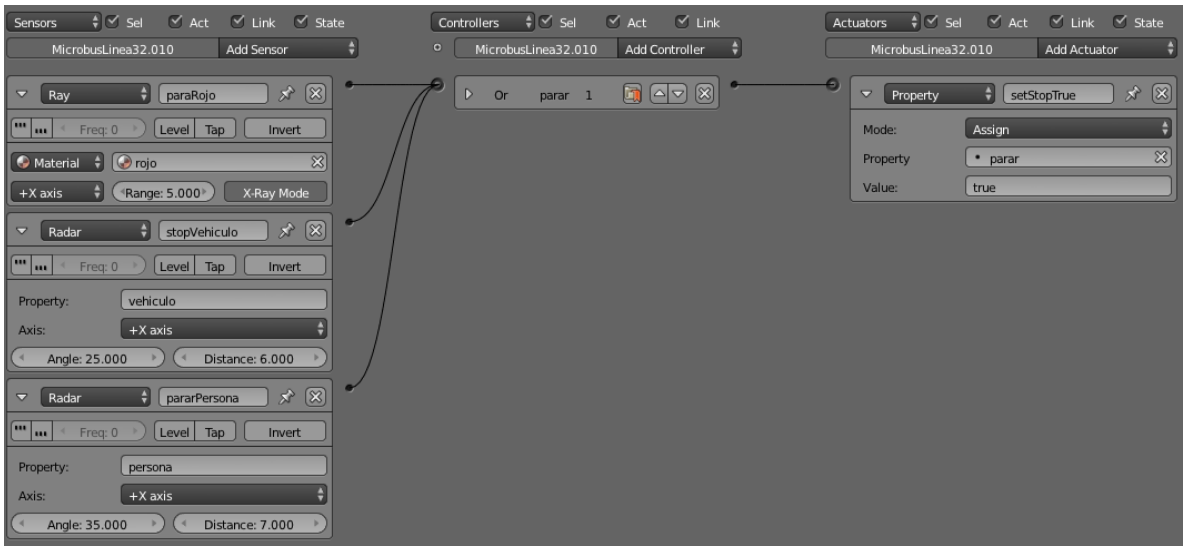


Figura lógica parar true

Lógica parar false

Se usa sensores para desactivar la propiedad de "parar", con el fin de volver a movilizar el vehículo mientras no se cumplan estas condiciones.

Se usa un controlador nor con el propósito de hacer valida la condición cuando todos los sensores son falsos.

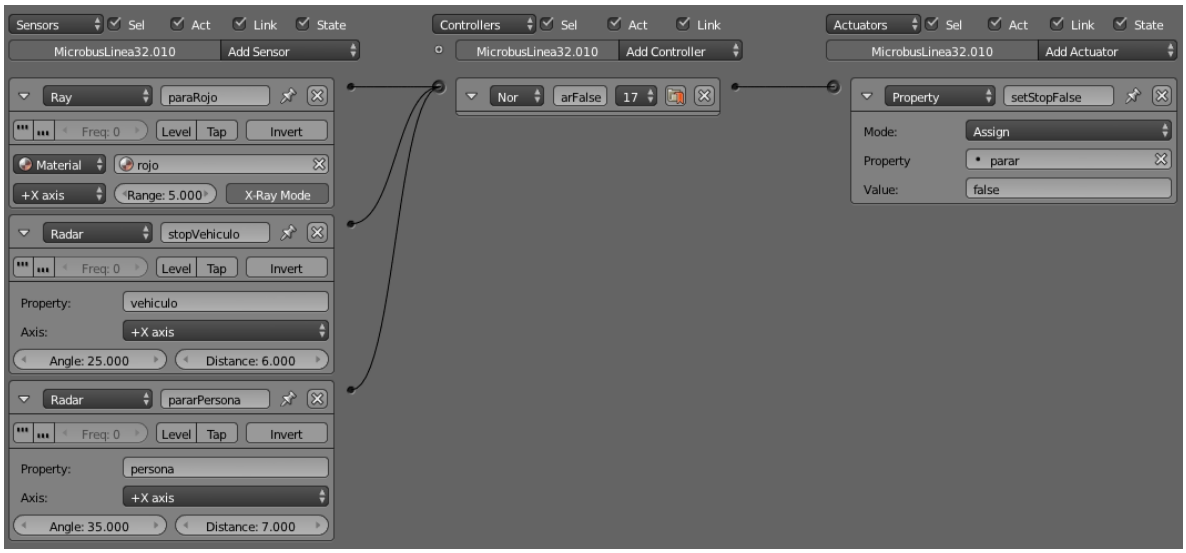


Figura lógica parar false

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica mover a objetivo normal.

Este actuador tiene por finalidad hacer que el objeto de mueva, utiliza el comportamiento integrado de Blender llamado "path following"[10], donde el objeto se dirige a un objetivo establecido, por medio de una malla de navegación a una velocidad determinada.

El utilizar la propiedad "facing" brinda al objeto dirigirse con una "cara" determinada, apuntando hacia el objetivo.

El tipo de controlador que se usa es de expresión, el cual permite ingresar como condiciones las propiedades del objeto.

Expression "frenar==false and parar ==false and choque==false".

El objeto se moverá a la una velocidad indicada, mientras no tenga que frenar, ni parar y tampoco haya sufrido un choque.

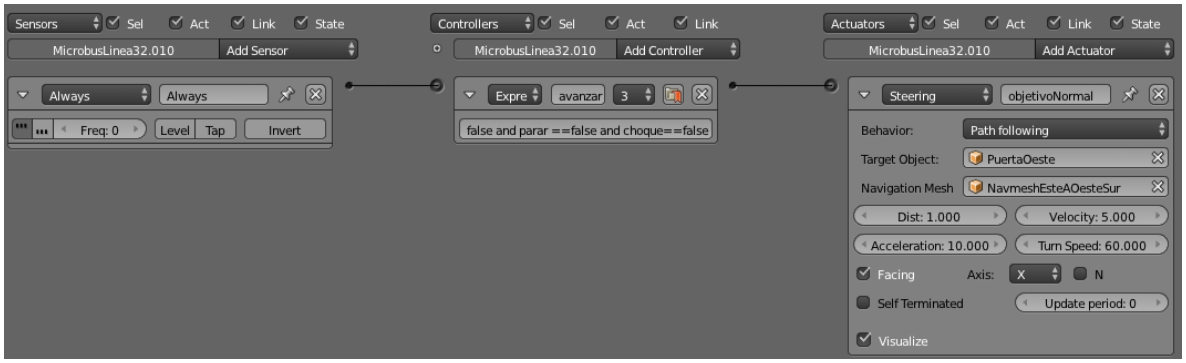


Figura lógica mover a objetivo normal.

[10] *http://www.tutorialsforblender3d.com/LogicBricks/Actuators/Steering/Steering_Actuator_Path_following.html*

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica mover a objetivo lento.

Este actuador tiene por finalidad hacer que el objeto de mueva lentamente, al igual que el sensor anterior utiliza el comportamiento integrado de Blender llamado "path following"[11], donde el objeto se dirige a un objetivo establecido, por medio de una malla de navegación a una velocidad determinada.

El tipo de controlador que se usa es de expresión, el cual permite ingresar como condiciones las propiedades del objeto.

Expression "frenar==true and parar ==false and choque==false".

El objeto se moverá a la una velocidad indicada como lenta, cuando tenga que frenar, pero no parar y tampoco haya sufrido un choque.

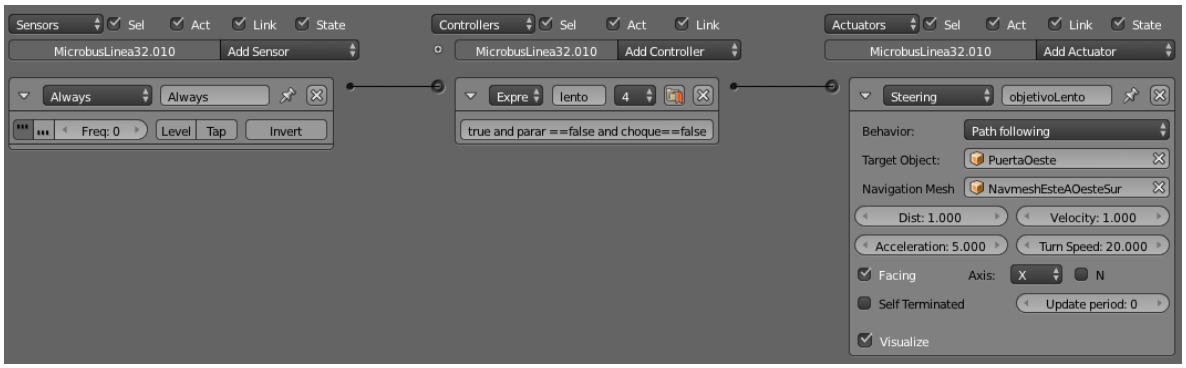


Figura lógica mover a objetivo lento.

[11] *http://www.tutorialsforblender3d.com/LogicBricks/Actuators/Steering/Steering_Actuator_Path_following.html*

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica choque.

Este conjunto de lógica es utilizado para establecer la variable choque verdadera, cuando el objeto choque con uno de estos tres tipos de objetos:

- Barreras que contengan el material verde_barrera.
- Objetos con propiedad vehículos.
- Objetos con propiedad persona.

Luego se establecer el choque como verdadero el objeto dejará de moverse.

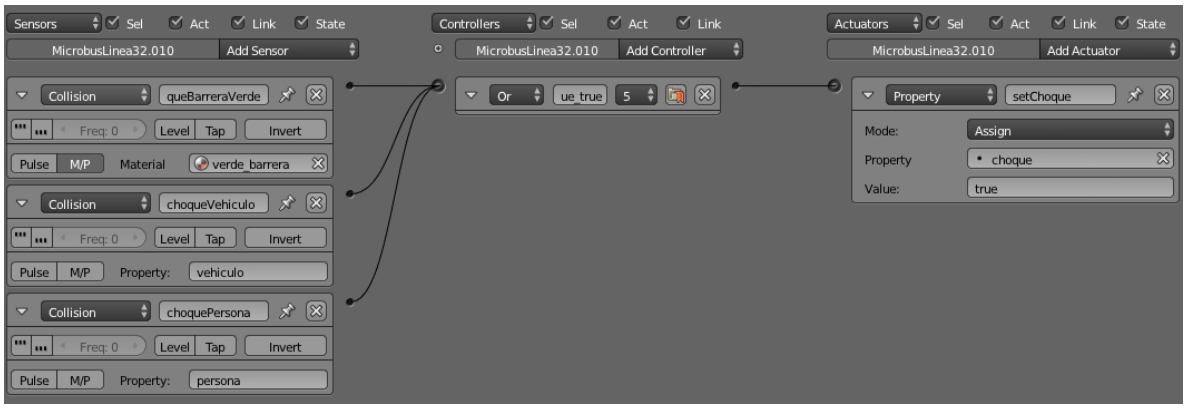


Figura sensor de choque.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica de frenado true

Sensores utilizados para activar la propiedad de frenar, con el fin de reducir la velocidad del vehículo, siempre y cuando se cumpla una de estas condiciones.

Detecte:

- Propiedad tipo persona en un objeto, a una distancia de 10 ub, es proyectada por el eje x positivo del objeto y con una amplitud de 50º.
- Propiedad tipo vehículo en un objeto, a una distancia de 10 ub, es proyectada por el eje x positivo del objeto y con una amplitud de 25º.
- Material tipo rojo en un objeto, dentro de un rango inferior o igual a 7 ub, y es proyectada por el eje x positivo del objeto.

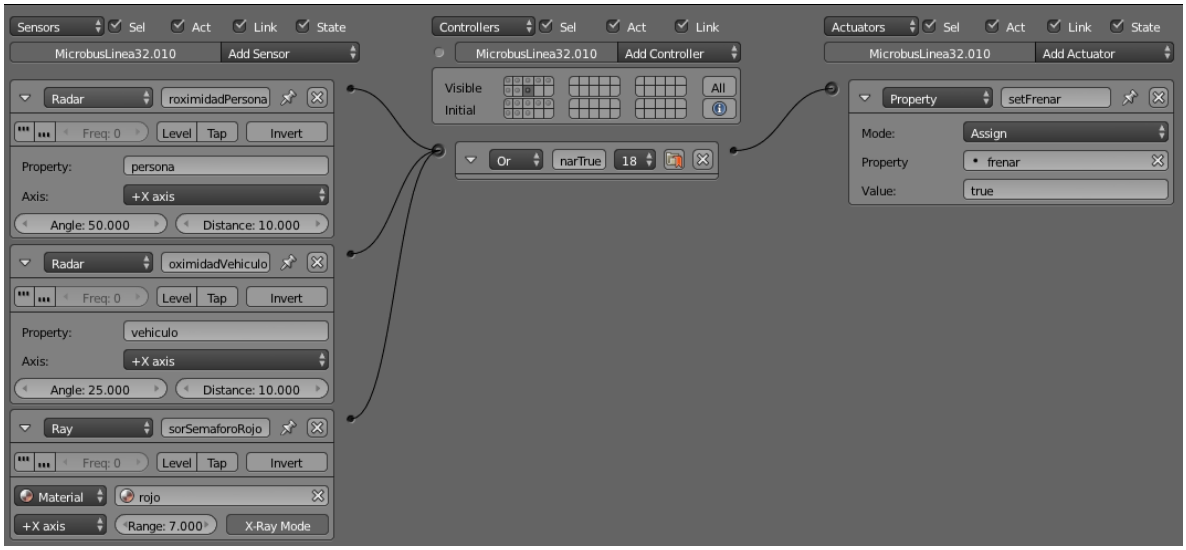


Figura lógica de frenado true.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica de frenado false

Sensores utilizados para desactivar la propiedad de frenar, mientras no se cumplan estas condiciones.

Se usa un controlador nor con el propósito de hacer valida la condición cuando todos los sensores son falsos.

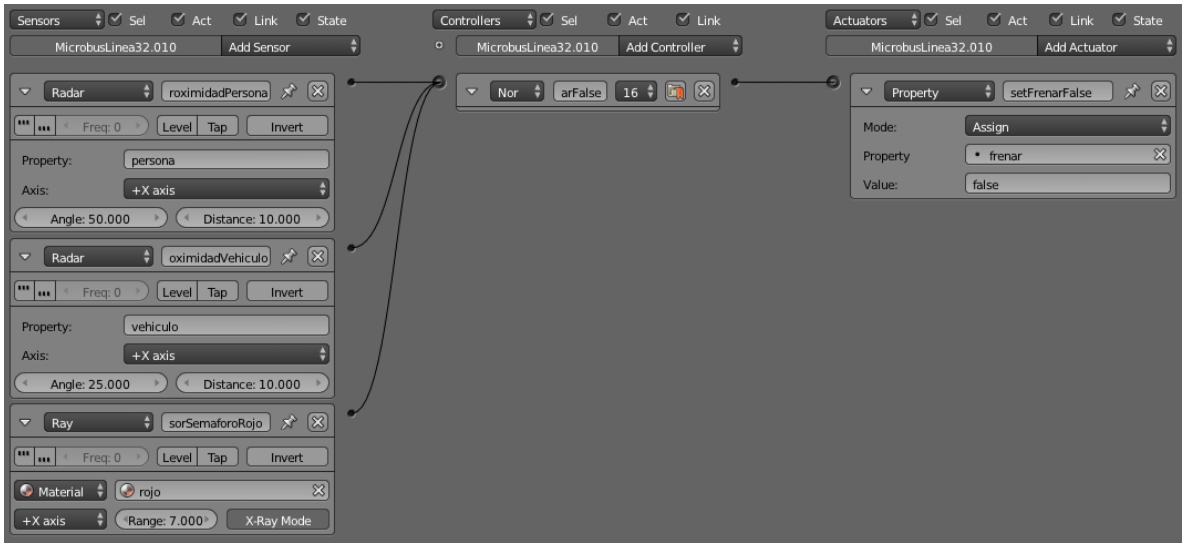


Figura sensor de frenado false.

Lógica reiniciar ubicación

Este sensor utiliza la colisión con un objeto tipo puerta el cual detecta y direcciona a través de un script, al objeto a otra ubicación descrita en este script.

Dicha ubicación se especifica en el controlador. Es así como en este caso Puerta.Collao2 indica que cuando exista una colisión con un objeto de tipo puerta, debe revisar el script "Puerta.py", y dentro donde se especifique "Collao2" y seguir sus instrucciones.

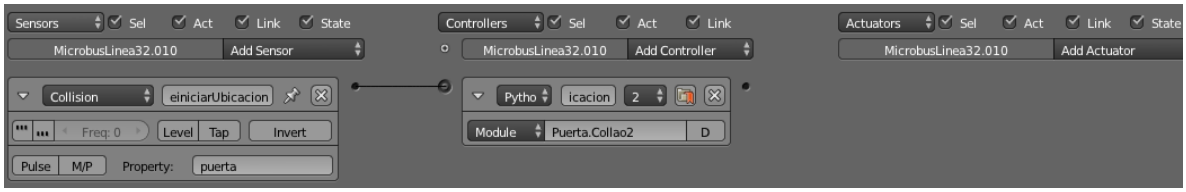


Figura sensor reiniciar ubicación

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.6.2. Componentes de humanos:

Propiedades para humanos

Las propiedades de los humanos sirven para identificar la acción que está realizando por ello:

- Choque, será positivo luego de que el vehículo tenga algún tipo de colisión detectada.
- La propiedad persona se utiliza para dar a conocer a los demás objetos el tipo al que pertenece.

Lógica de acciones para humanos:

Lógica reiniciar ubicación

Este sensor al igual al que poseen los vehículos, utiliza la colisión con un objeto tipo puerta el cual detecta y direcciona a través de un script, al objeto a otra ubicación descrita en este script.

Dicha ubicación se especifica en el controlador. Es así como en este caso "Puerta.PeatonIrarrazaval1", indica que cuando exista una colisión con un objeto de tipo puerta, debe revisar el script "Puerta.py", y dentro donde se especifique "PeatonIrarrazaval1" y seguir sus instrucciones.

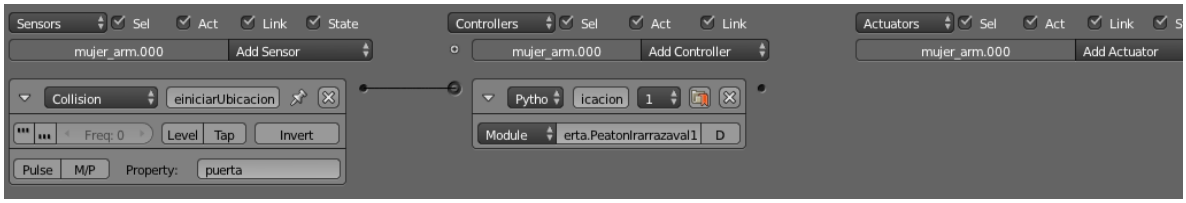


Figura sensor reiniciar ubicación

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica avanzar.

Este actuador al igual que el que hace mover los objetos tipo vehículo, tiene por finalidad hacer que el objeto de mueva, utiliza el comportamiento integrado de Blender llamado "path following"[12], donde el objeto se dirige a un objetivo establecido, por medio de una malla de navegación a una velocidad determinada.

El tipo de controlador que se usa es de expresión, el cual permite ingresar como condiciones las propiedades del objeto.

Expression "frenoSemaforo==false and choque==false and Always == true".

El objeto se moverá a la una velocidad indicada, mientras no tenga que parar y tampoco haya sufrido un choque.

El controlador acciona además a un actuador de tipo action, en donde se selecciona la animación walk.

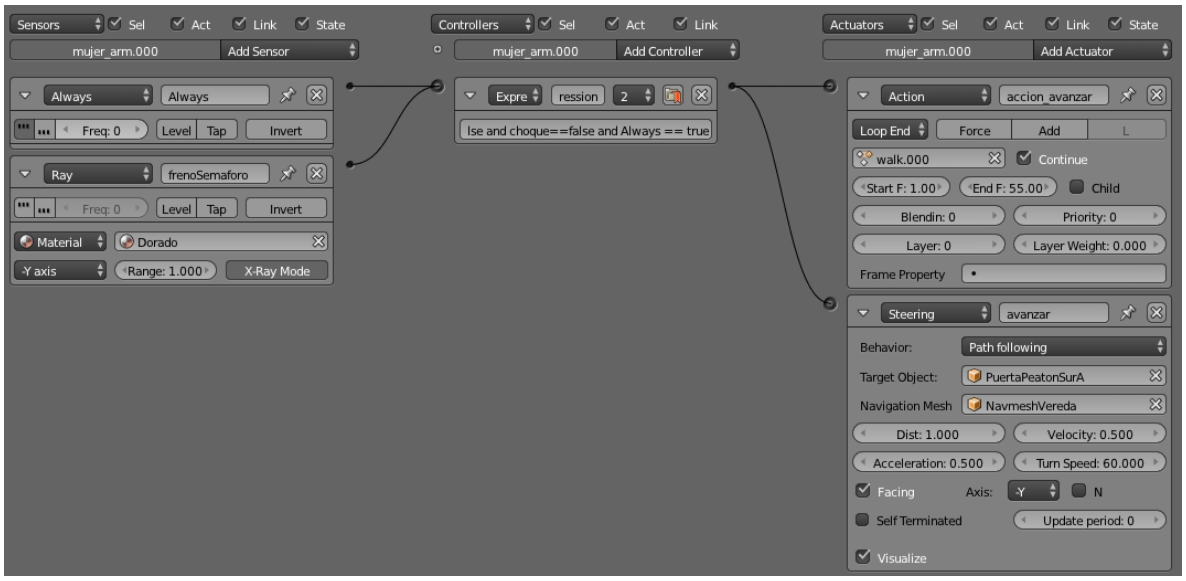


Figura lógica acción avanzar.

[12] *http://www.tutorialsforblender3d.com/LogicBricks/Actuators/Steering/Steering_Actuator_Path_following.html*

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica choque.

Este conjunto de lógica es utilizado para establecer la variable choque verdadera, cuando el objeto choque con objetos con propiedad vehículo.

Luego se establecer el choque como verdadero y el objeto dejará de avanzar.

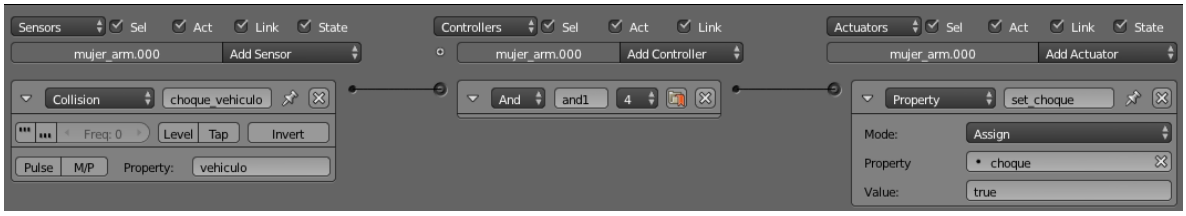


Figura sensor de Choque.

Lógica de detención.

Sensores utilizados detener al objeto, mientras se cumpla una de estas condiciones.

Detecte:

- Material tipo azul/dorado en un objeto, dentro de un rango inferior o igual a 1 ub, y es proyectada por el eje y negativo del objeto del objeto.
- Propiedad tipo vehículo en un objeto, a una distancia de 3 ub, es proyectada por el eje y negativo del objeto y con una amplitud de 20º.

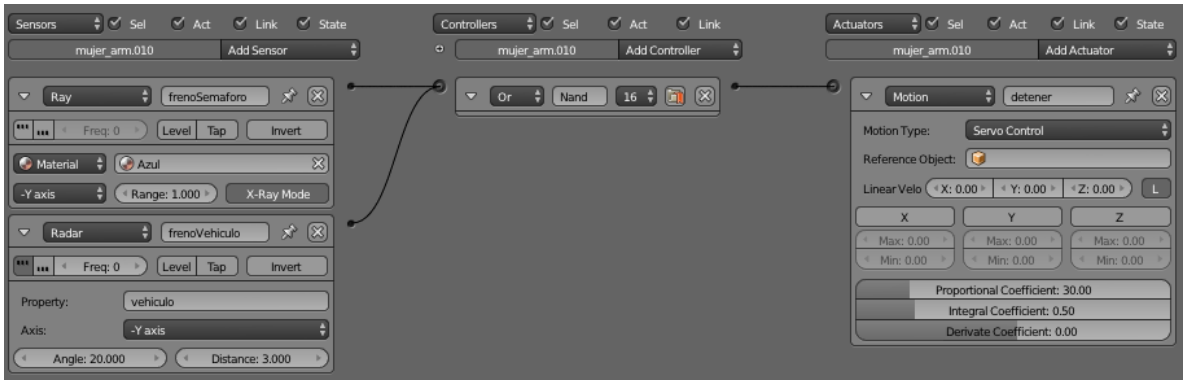


Figura lógica acción parar.

La variación en cuando al color de material que detectan, se realiza debido a que un peatón al cruzar la calle no obedece al mismo semáforo, por esta razón se debe de establecer por ejemplo que un peatón que cruza de un sector A detecte sólo colores azul, mientras tanto otro peatón que cruza de un sector B respeta colores dorado.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

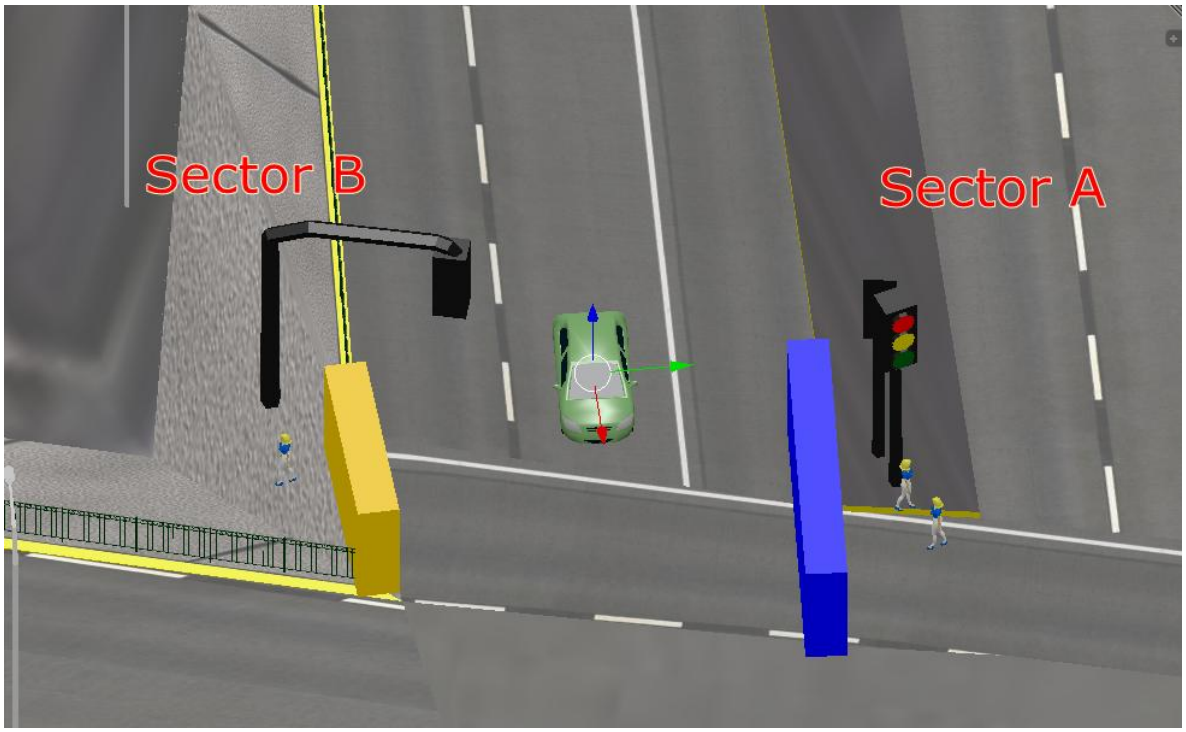


Figura ejemplo barreras semáforos peatón.

Todo esto con el propósito de evitar que el peatón quede detenido en medio de la calzada

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.6.3. Componentes de semáforos

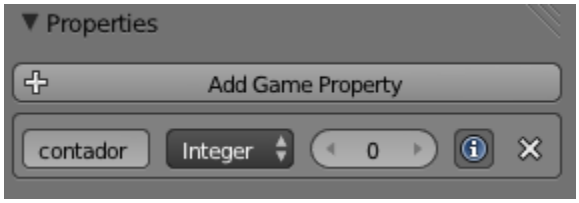
El semáforo es un actor compuesto por un conjunto de objetos. Existe un semáforo visual que brinda la sensación de realidad al observador, pero su objetivo sólo tiene que ver con ser visualizado, y por otra parte existe el semáforo funcional, el cual cumple el objetivo de controlar el tránsito de los objetos que lo detectan.

- Semáforo visual: semáforo piso, aéreo y similares que contengan los colores correspondientes.
- Semáforo funcional: barrera semáforo tiene forma rectangular y sirve para ser detectado por otros objetos.

Propiedades

Las propiedades de los semáforos sirven como variables y en este caso sólo se necesita una:

- Contador, simula el segundero de un reloj aumentando aproximadamente una unidad por segundo permitiendo así, controlar la duración de cada uno de los estados del semáforo.



Lógica de acciones para semáforos:

La lógica para las acciones varía en cuanto a la condiciones del orden de aparición dado que los colores tienen una secuencia de aparición pero basta con establecer el valor correctamente para que se visualice de acuerdo a lo esperado.

Se usará como ejemplo un semáforo de 47 segundos.

Lógica semáforo color verde visible

Para que sea visible el color verde del semáforo se crea una condición lógica que permite su visibilidad cuando el contador sea igual a 0.

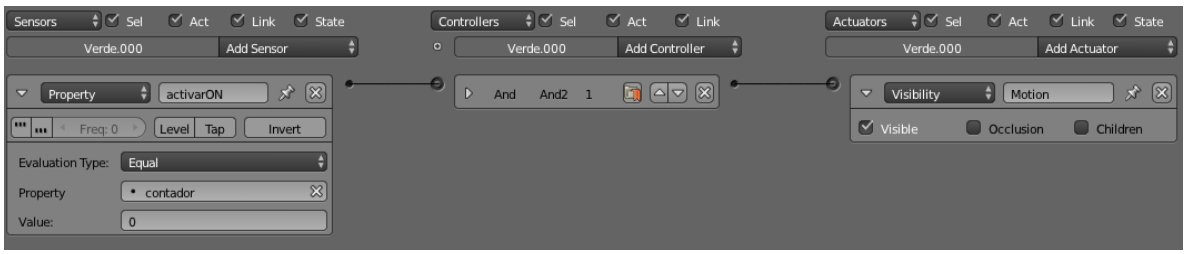


Figura lógica semáforo visible

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica semáforo color verde invisible

Para que sea invisible el color verde del semáforo se crea una condición lógica que permite su invisibilidad cuando el contador sea igual a 8.

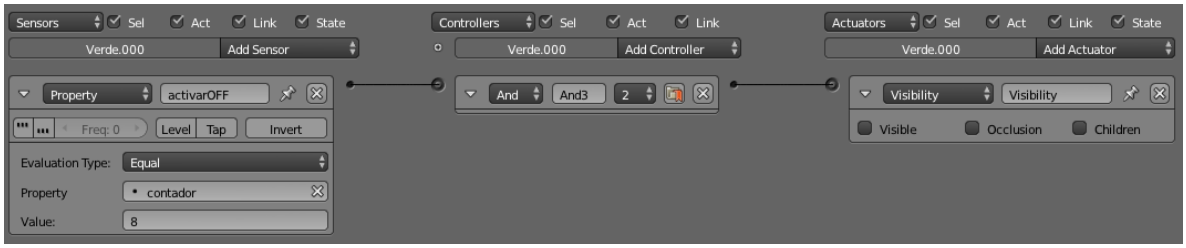


Figura lógica semáforo invisible

Lógica semáforo aumentar contador

Al utilizar la propiedad contador como timer, es necesario aumentarlo secuencialmente, esto se realiza usando un sensor delay y estableciendo el retraso en 60 fps, lo que equivale a 1 unidad por segundo, además se aplica el atributo tap, forzando que reenvíe constantemente la señal positiva.

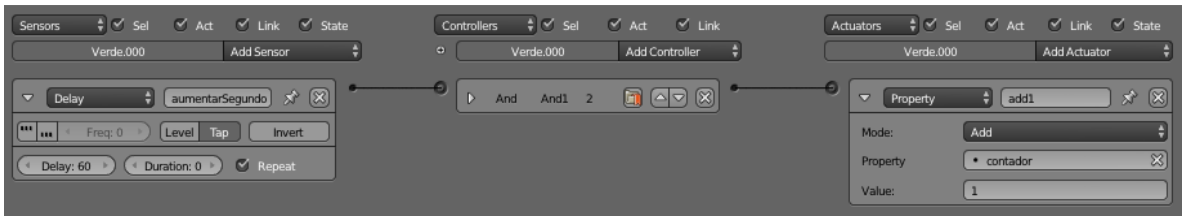


Figura lógica semáforo aumentar contador

Lógica semáforo restablecer contador

Al utilizar la propiedad contador como timer, es necesario volver a iniciar el contador en 0, formando así un ciclo para el semáforo, en este caso cuando el valor del contador es igual a 47 el contador se establece en 0.

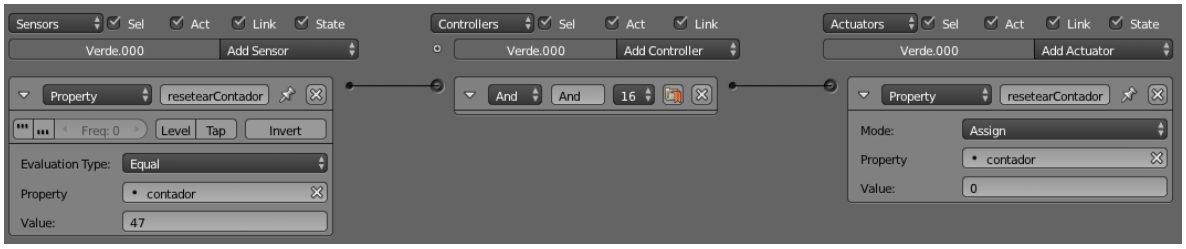


Figura lógica semáforo restablecer contador

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica semáforo color amarillo visible

Para que sea visible el color amarillo del semáforo se crea una condición lógica que permite su visibilidad cuando el contador sea igual a 8.

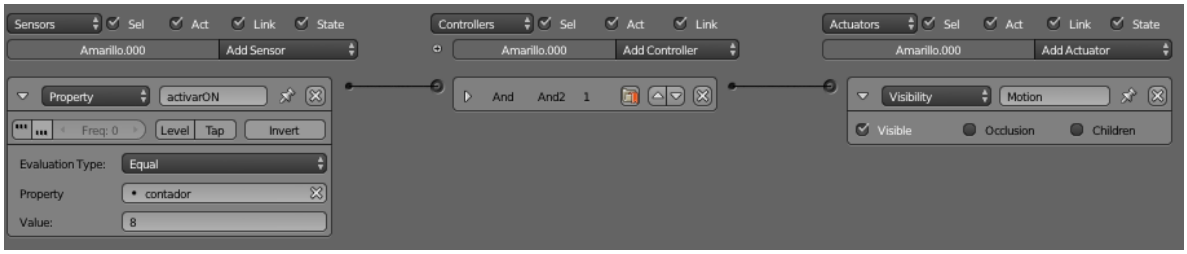


Figura lógica semáforo visible

Lógica semáforo color amarillo invisible

Para que sea invisible el color amarillo del semáforo se crea una condición lógica que permite su invisibilidad cuando el contador sea igual a 10.

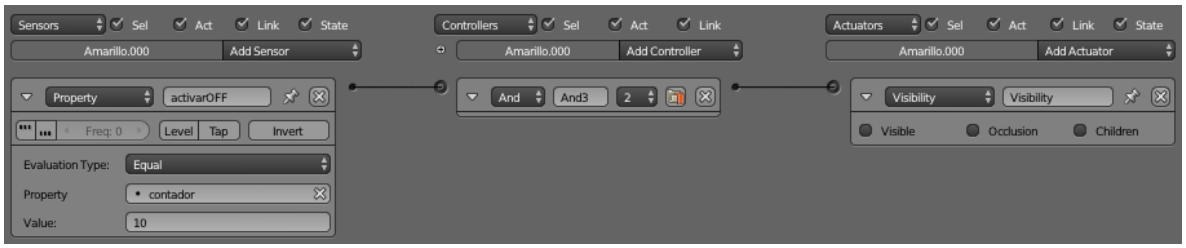


Figura lógica semáforo invisible

Lógica semáforo aumentar contador

Al utilizar la propiedad contador como timer, es necesario aumentarlo secuencialmente, esto se realiza usando un sensor delay y estableciendo el retraso en 60 fps, lo que equivale a 1 unidad por segundo, además se aplica el atributo tap, forzando que reenvíe constantemente la señal positiva.

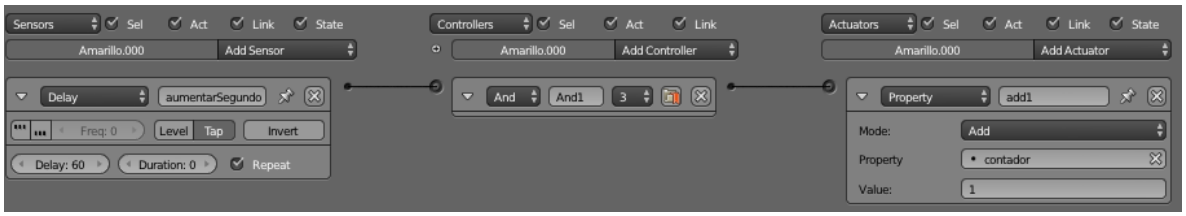


Figura lógica semáforo aumentar contador

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica semáforo restablecer contador

Al utilizar la propiedad contador como timer, es necesario volver a iniciar el contador en 0, formando así un ciclo para el semáforo, en este caso cuando el valor del contador es igual a 47 el contador se establece en 0.

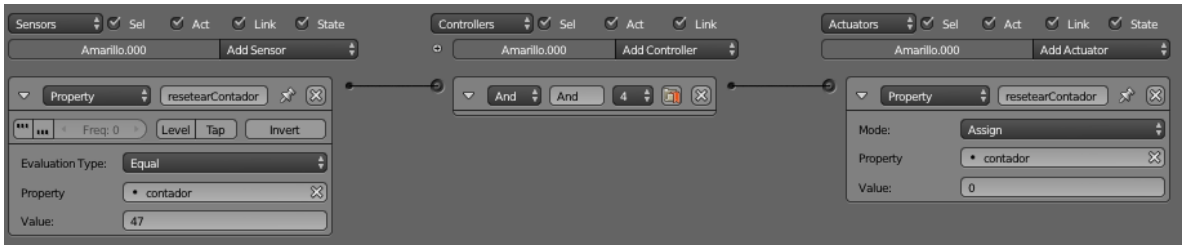


Figura lógica semáforo restablecer contador

Lógica semáforo color rojo visible

Para que sea visible el color rojo del semáforo se crea una condición lógica que permite su visibilidad mientras el contador sea igual a 10, la manera más simple para guiarse en cuanto a la configuración que tendrá es usar los valores del semáforo en verde más el amarillo e invertirlos por ello si se conoce que desde el numero 0 a 8 será verde y luego que de 8 a 10 será amarillo se sabe que el color rojo debe aparecer luego del amarillo o sea en 10 y finalizar en 0, cuando el color verde vuelva a aparecer.

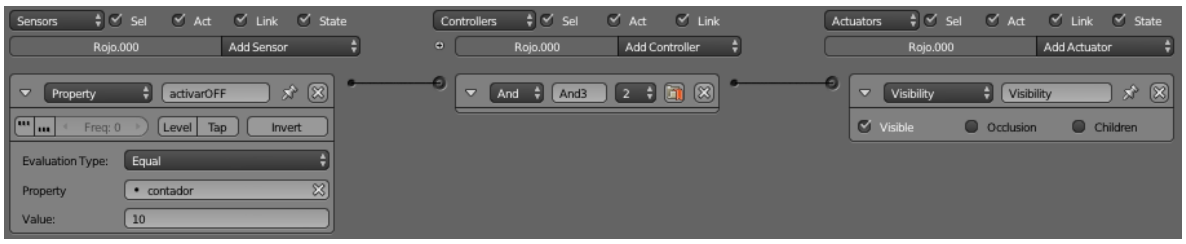


Figura lógica semáforo visible

Lógica semáforo color rojo invisible

Para que sea invisible el color rojo del semáforo se crea una condición lógica que permite su invisibilidad cuando el contador sea igual a 0.

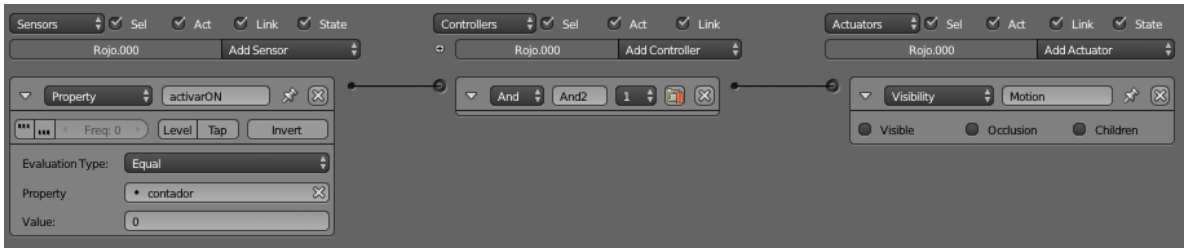


Figura lógica semáforo invisible

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica semáforo aumentar contador

Al utilizar la propiedad contador como timer, es necesario aumentarlo secuencialmente, esto se realiza usando un sensor delay y estableciendo el retraso en 60 fps, lo que equivale a 1 unidad por segundo, además se aplica el atributo tap, forzando que reenvíe constantemente la señal positiva.

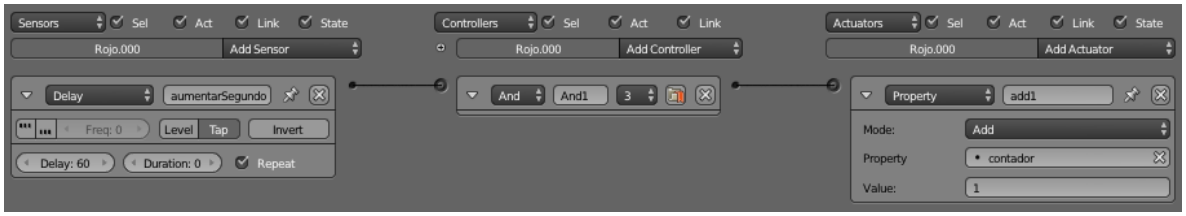


Figura lógica semáforo aumentar contador

Lógica semáforo restablecer contador

Al utilizar la propiedad como timer es necesario volver a iniciar el contador en 0, formando así un ciclo para el semáforo, en este caso cuando el valor del contador es igual a 47 el contador se establece en 0.

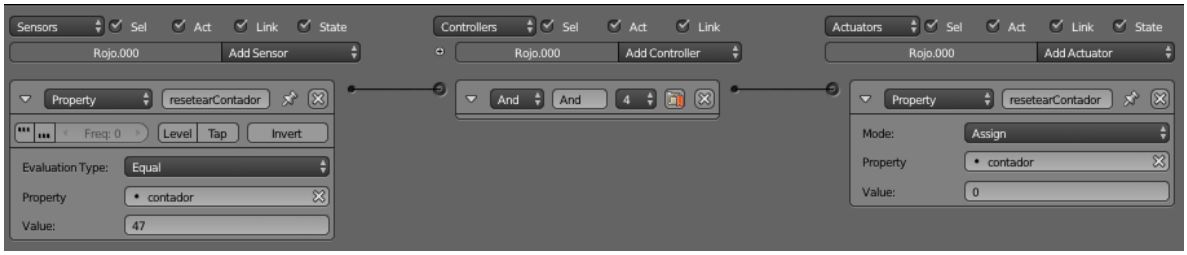


Figura lógica semáforo restablecer contador

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica barrera semáforo on

Para que sea visible el objeto se crea una condición lógica que permite su aparición en la calzada por medio de un script escrito en lenguaje Python, en este caso, cuando el contador sea igual a 10.

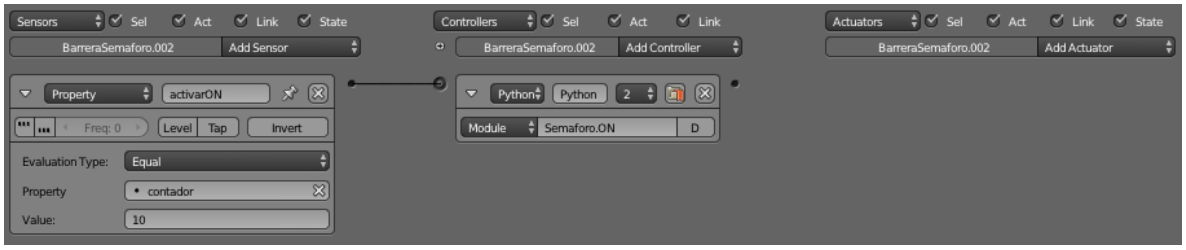


Figura lógica barrera semáforo on

Lógica barrera semáforo off

Para que no sea visible el objeto se crea una condición lógica que permite su desaparición de la calzada por medio de un script escrito en lenguaje Python, en este caso, cuando el contador sea igual a 0.

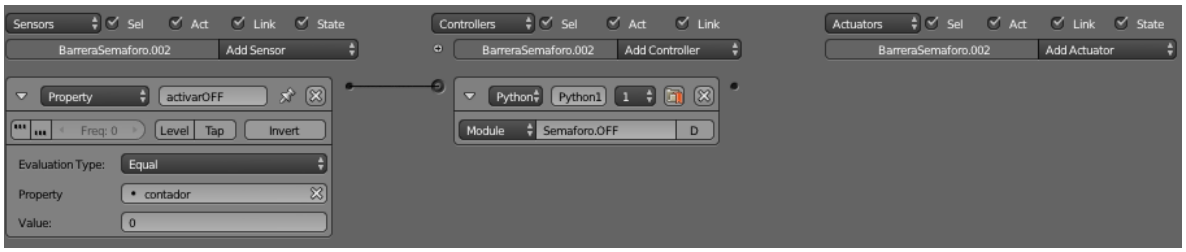


Figura lógica barrera semáforo off

Lógica semáforo aumentar contador

Al igual que el semáforo visual, se utiliza la propiedad contador como timer y es necesario aumentarlo secuencialmente, esto se realiza usando un sensor delay y estableciendo el retraso en 60 fps, lo que equivale a 1 unidad por segundo, además se aplica el atributo tap, forzando que reenvíe constantemente la señal positiva.

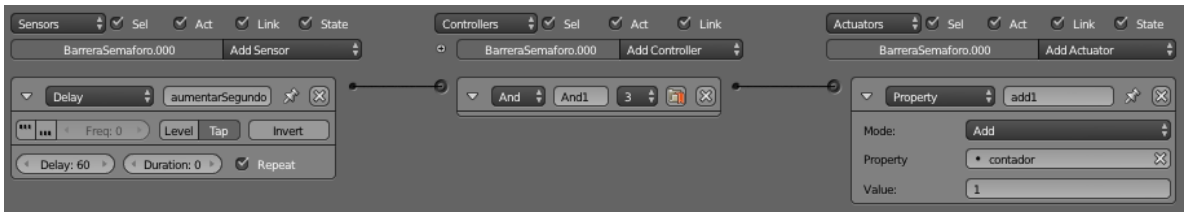


Figura lógica semáforo aumentar contador

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica semáforo restablecer contador

Al igual que el semáforo visual, al utilizar la propiedad contador como timer es necesario volver a iniciar el contador en 0, formando así un ciclo para el semáforo, en este caso cuando el valor del contador es igual a 47 el contador se establece en 0.

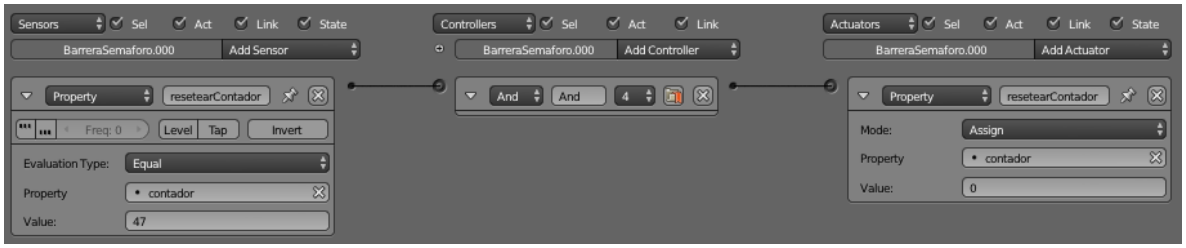


Figura lógica semáforo restablecer contador

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.6.4. Semáforos combinados

Para poder lograr una correcta simulación del semáforo, es importante sincronizarlos de forma apropiada.

Esto es que al aparecer el color rojo en el semáforo visual, debe de aparecer el semáforo funcional (barrera semáforo) en medio de la calzada permitiendo ser detectado por los objetos apropiados vehículos o peatones dependiendo del caso.

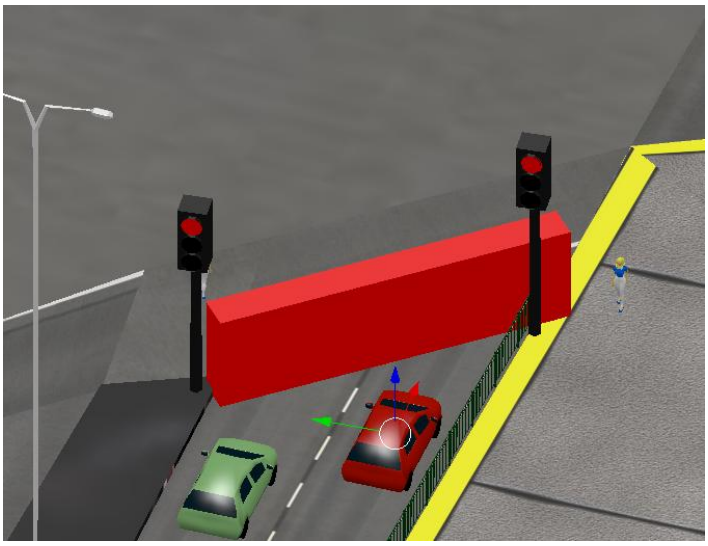


Figura semáforo rojo

En la figura se aprecia cómo debe ser el resultado de una buena sincronía entre ambos semáforos para el caso de ser rojo (impedir el tránsito).

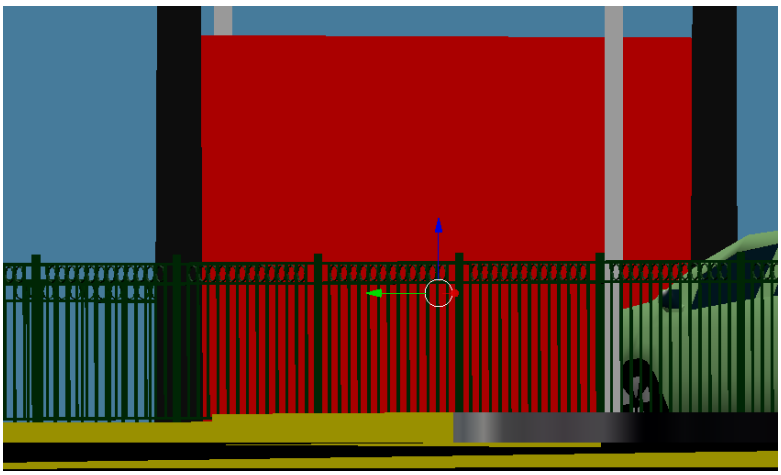


Figura semáforo rojo vista lateral

En esta figura se ve la posición de la barrera roja, que cumple función de semáforo, debe quedar posicionada sobre la calzada para poder ser detectada por los vehículos.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

En el caso de aparecer el color verde, condición que permite el tránsito de vehículos o personas dependiendo de la situación, la barrera semáforo no debe de aparecer en la calzada

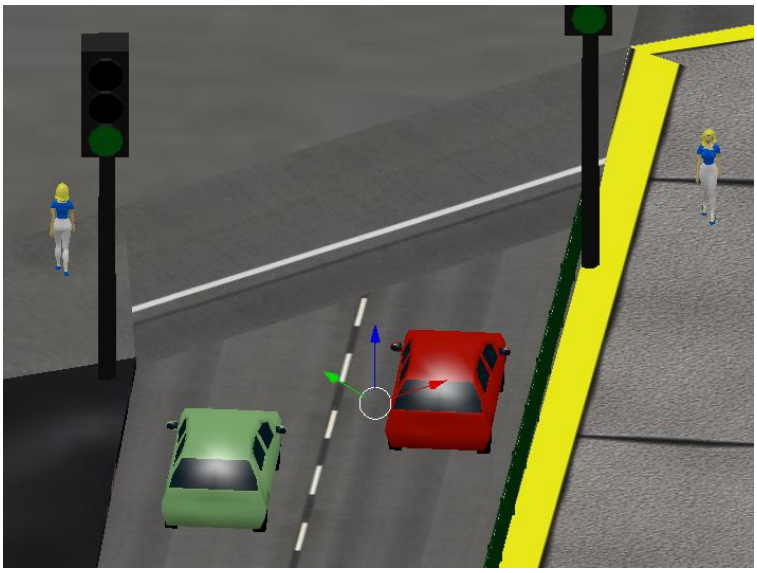


Figura semáforo verde

En la figura se aprecia cómo debe ser el resultado de una buena sincronía entre ambos semáforos para el caso de ser verde (permitir el tránsito).

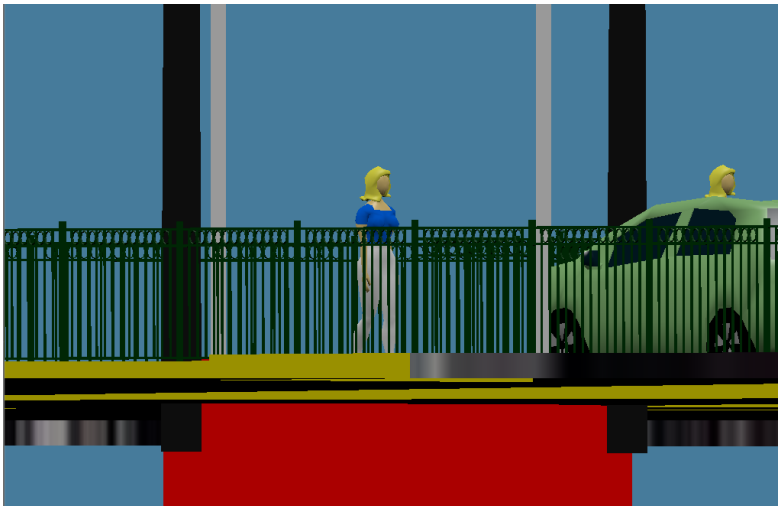


Figura semáforo verde vista lateral

En la figura se logra apreciar el funcionamiento de la barrera semáforo. Cuando la condición es de permitir el tránsito, ésta se desplaza -4 ub, por el eje z, siendo indetectable para los actores.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.7. Construir el escenario virtual a partir de los objetos.

Finalmente, se construye el escenario virtual a partir de los objetos en conjunto.

Desarrollo Intersección Collao, Los Carrera e Irarrázaval.

7.7.1. Incorporación objetos

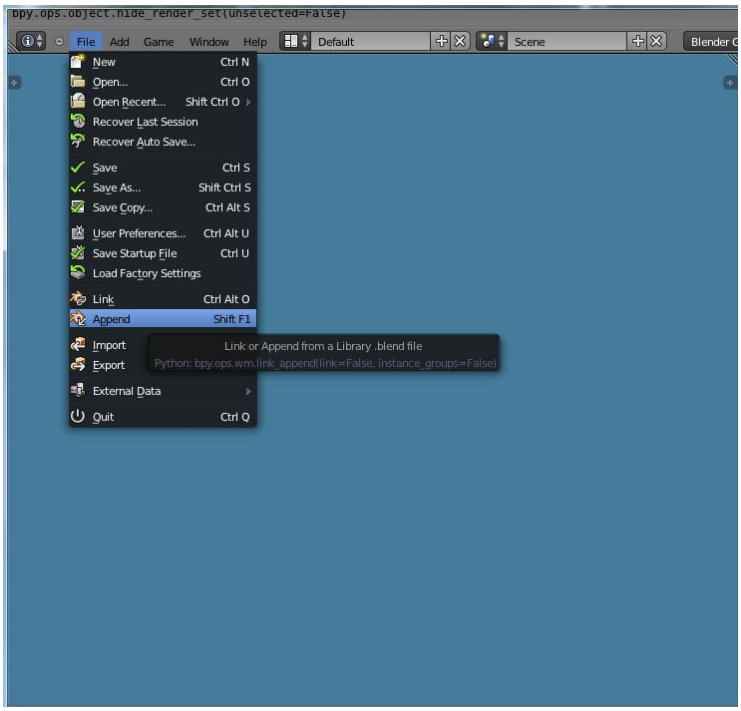


Figura incorporación de objetos

Para poder utilizar los objetos ya construidos en Blender, se deben de incorporar mediante el menú file opción append, luego se indica la ubicación del archivo del objeto, y se incorpora ubicándolo en la posición adecuada.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Una vez ubicados los objetos se duplican en el caso de ser necesario.

Para poder duplicar objetos en Blender, se utiliza el atajo de teclado shift+D, luego de haberlo seleccionado.

Otra forma de poder reutilizar las acciones lógicas de objetos, con la finalidad de no volver a programarlos es unir dos objetos distintos.

Primero se selecciona el objeto que no tiene lógica de acciones (ejemplo automóvil).

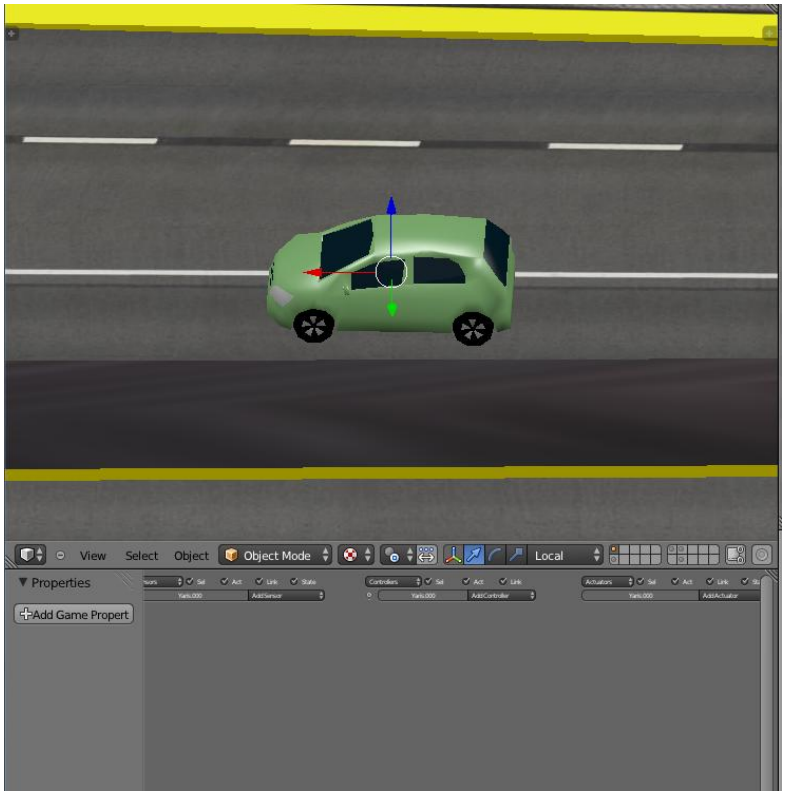


Figura objeto sin lógica de comportamiento

Luego manteniendo seleccionado el anterior (con shift), selecciona el objeto que posee lógica (en este caso el microbús)

Universidad del Bío-Bío. Red de Bibliotecas - Chile

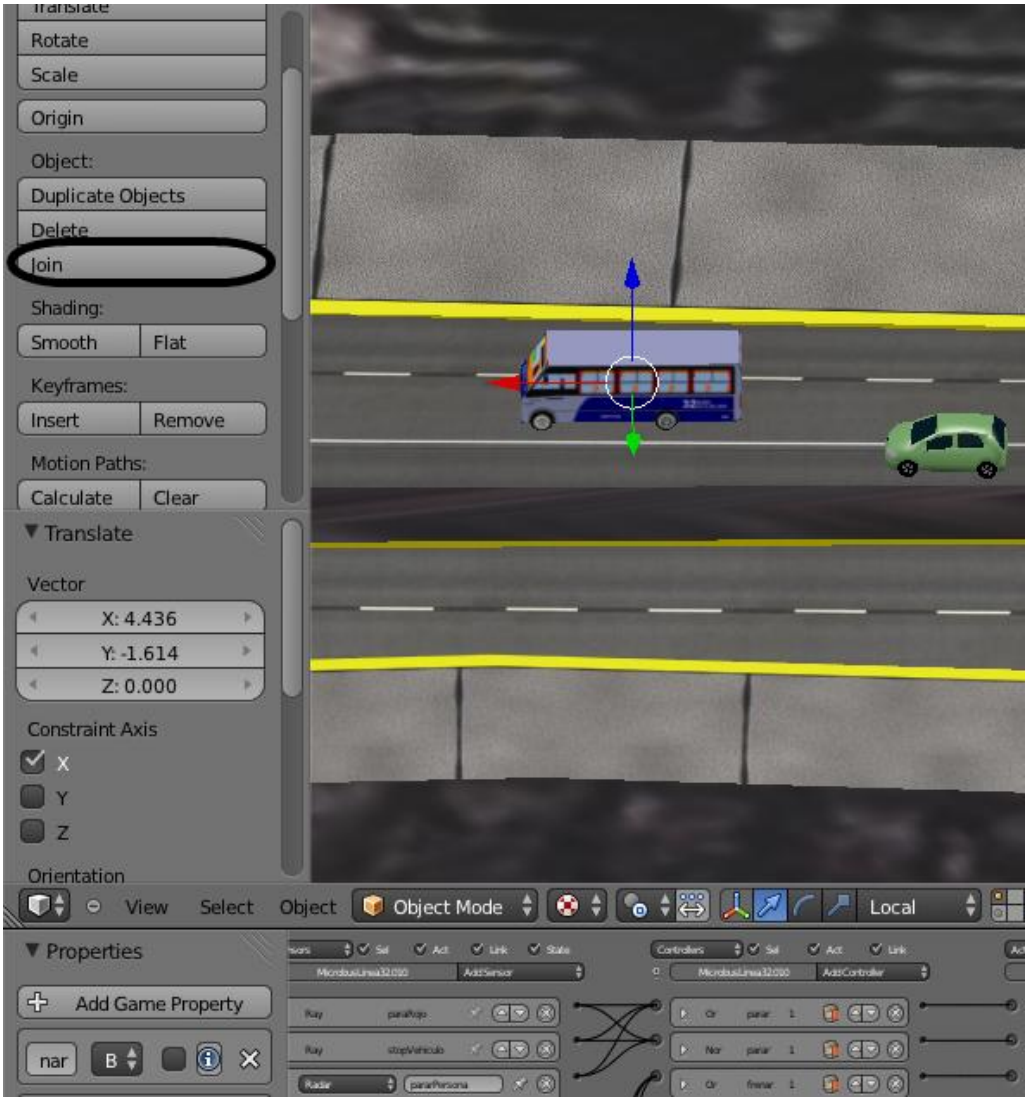


Figura unión de objetos

En la figura, se observa que al clicar sobre join se pueden unir ambos objetos, otra forma es a través del atajo de teclado ctrl + J.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

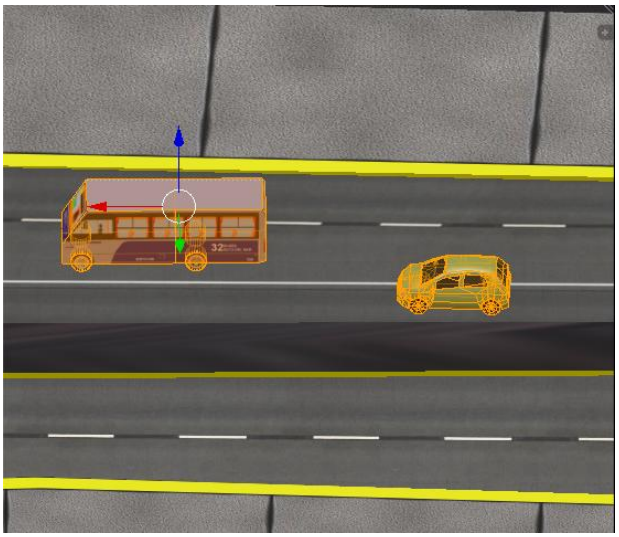


Figura resultado de unir

En esta figura se observa que ambos objetos se unieron y su referencia fue el microbús, por esto posee el punto de origen dicho objeto

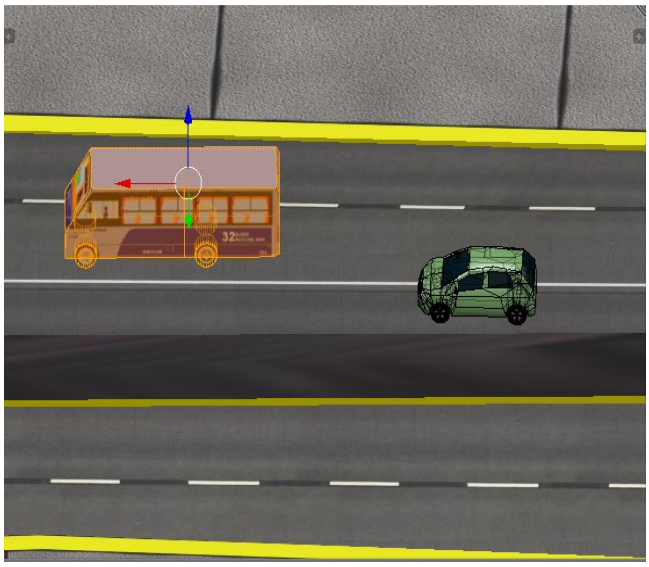


Figura selección objeto

Luego de la unión, se debe seleccionar el objeto para volver a separarlos, una manera simple es posar el cursor sobre el objeto y presionar la tecla "L", en el teclado.

Es importante verificar que este correctamente seleccionado el objeto deseado.

Una vez seleccionado el objeto que se requiere ser separado, se debe presionar la tecla "P"(siempre en el modo de edición).

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Figura resultado final esperado.

Este el resultado final esperado, luego de todo el proceso, traspasar la lógica de un objeto ya creado y probado, a otro nuevo, evitando reescribir código además de la lógica de acciones.

Observación

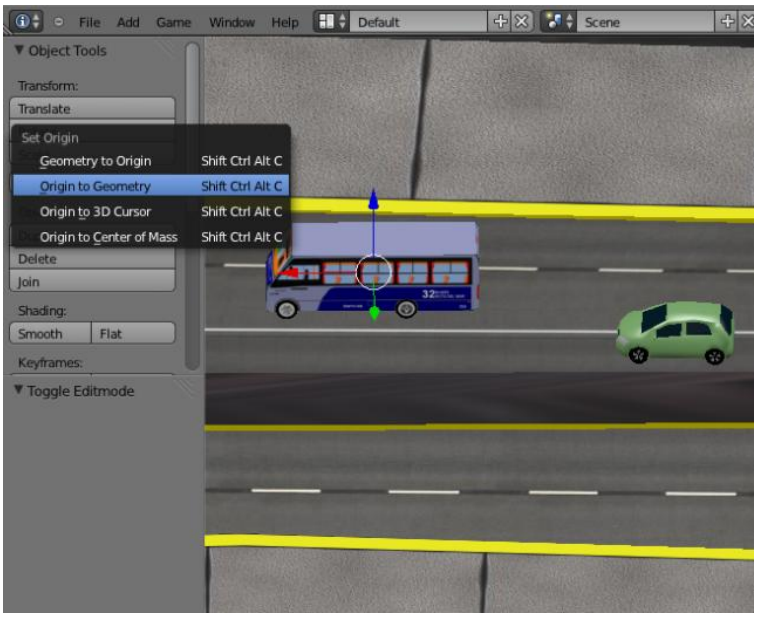


Figura corrección punto de origen.

También, es recomendable modificar el origen del actor, a través de las herramientas de objeto a la izquierda, para poder trabajar de una mejor forma posteriormente.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Utilizando la duplicación de objetos, copia de lógicas de acciones y ubicando adecuadamente los objetos se llega a este resultado

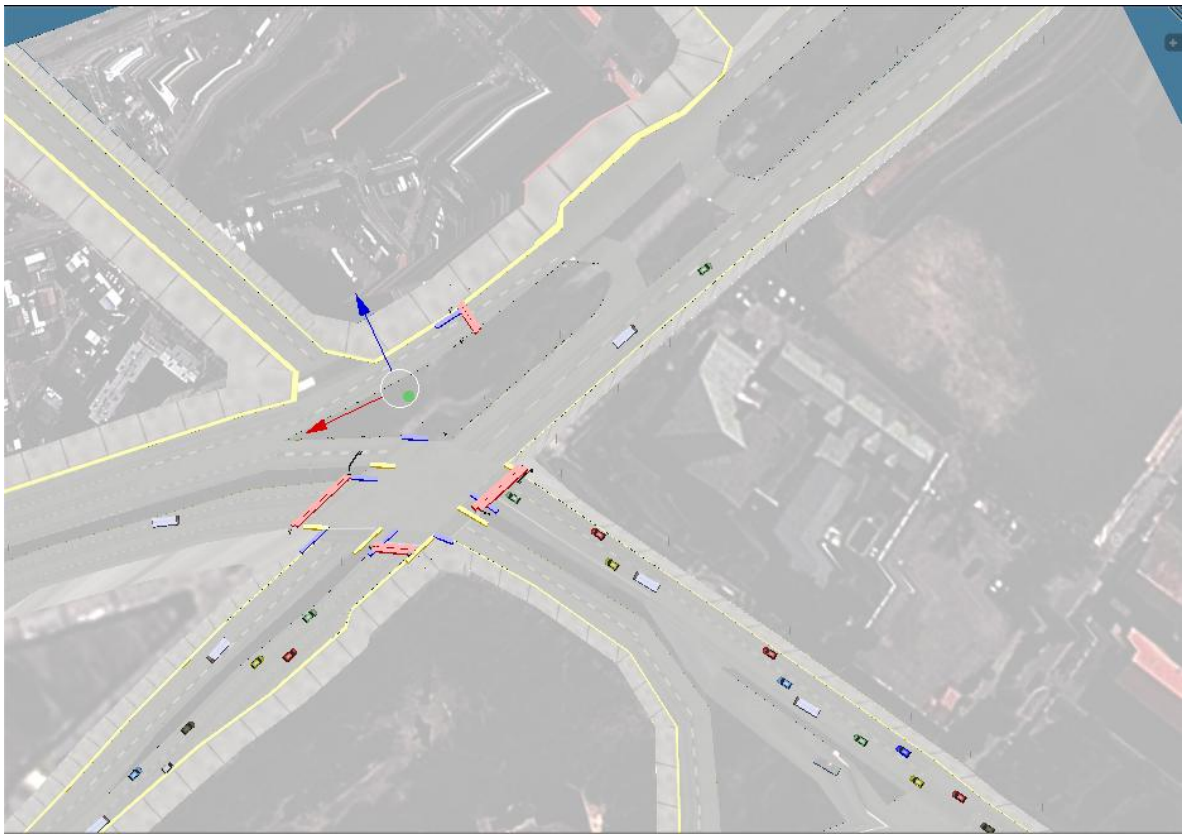


Figura ubicación de objetos.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Una vez ubicados los objetos, hay que establecer los parámetros que correspondan a cada uno, direccionando a aquellos que deban cumplir con dicha característica, como los peatones y vehículos que circulen. Además de configurar correctamente los semáforos evitando choques entre los actores.

7.7.2. Movimiento de vehículos

Lo primero que se debe hacer es establecer el objetivo o meta de cada vehículo, para esto se fijan las puertas como objetivos en este caso quedando estipuladas así.

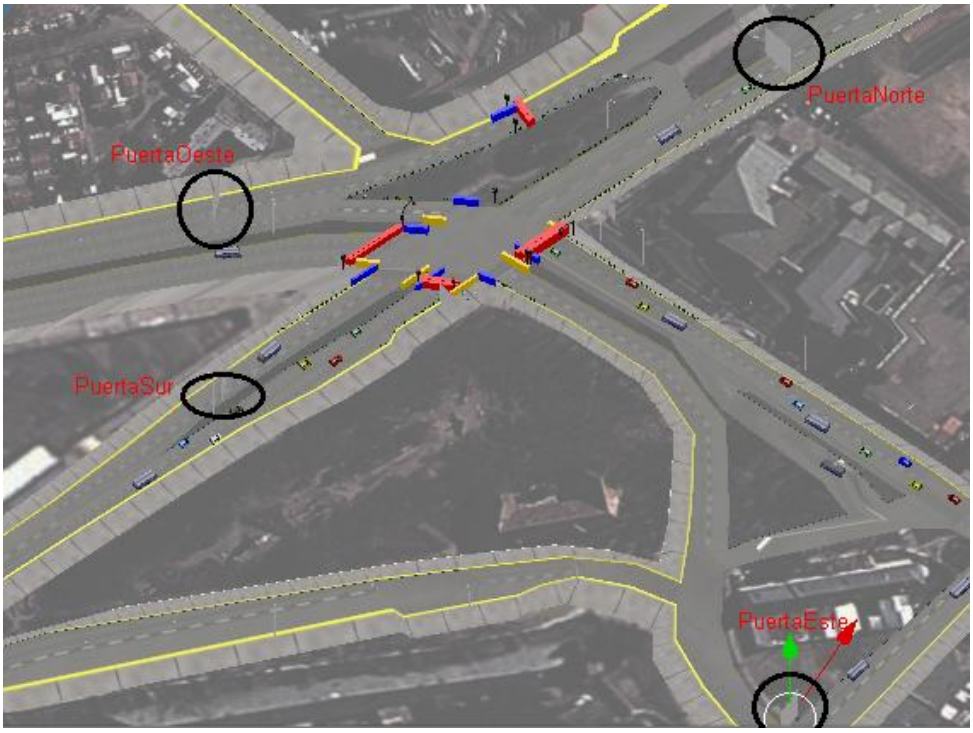


Figura ubicación de puertas objetivo para vehículos

Así, vehículos que circulan desde avenida Collao pueden optar por tres destinos PuertaOeste, PuertaNorte o PuertaSur.

Para fijar el destino apropiado se debe establecer cambiando la propiedad Target Object, fijando el objeto adecuado, además de fijar la malla de navegación correspondiente de la siguiente forma:

Universidad del Bío-Bío. Red de Bibliotecas - Chile

PuertaOeste

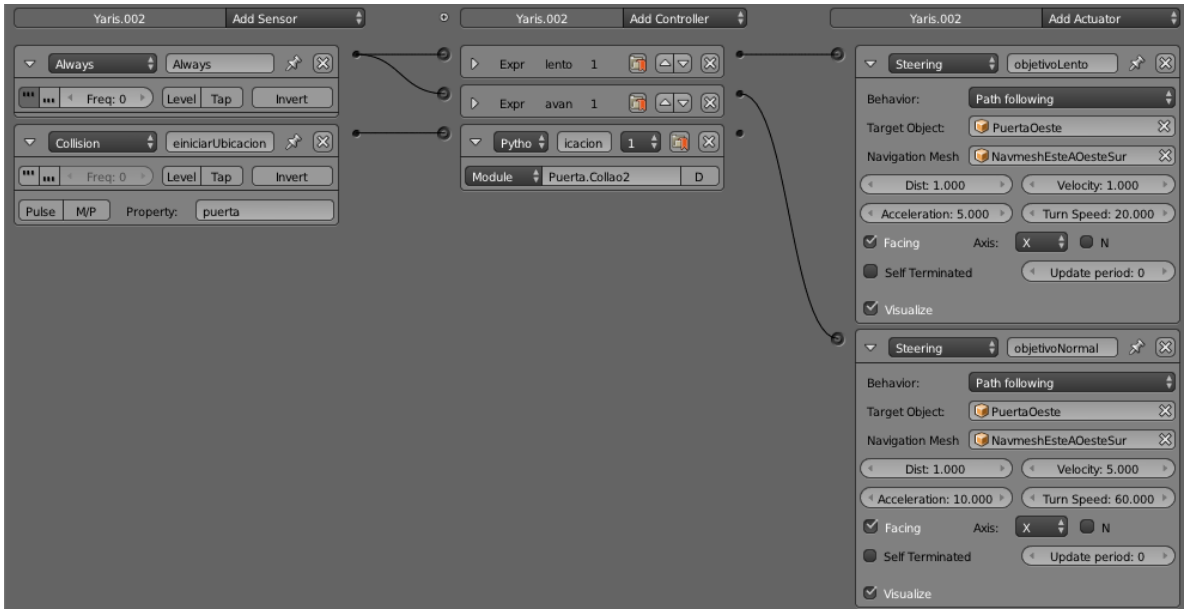


Figura lógica movimiento destino PuertaOeste

PuertaNorte

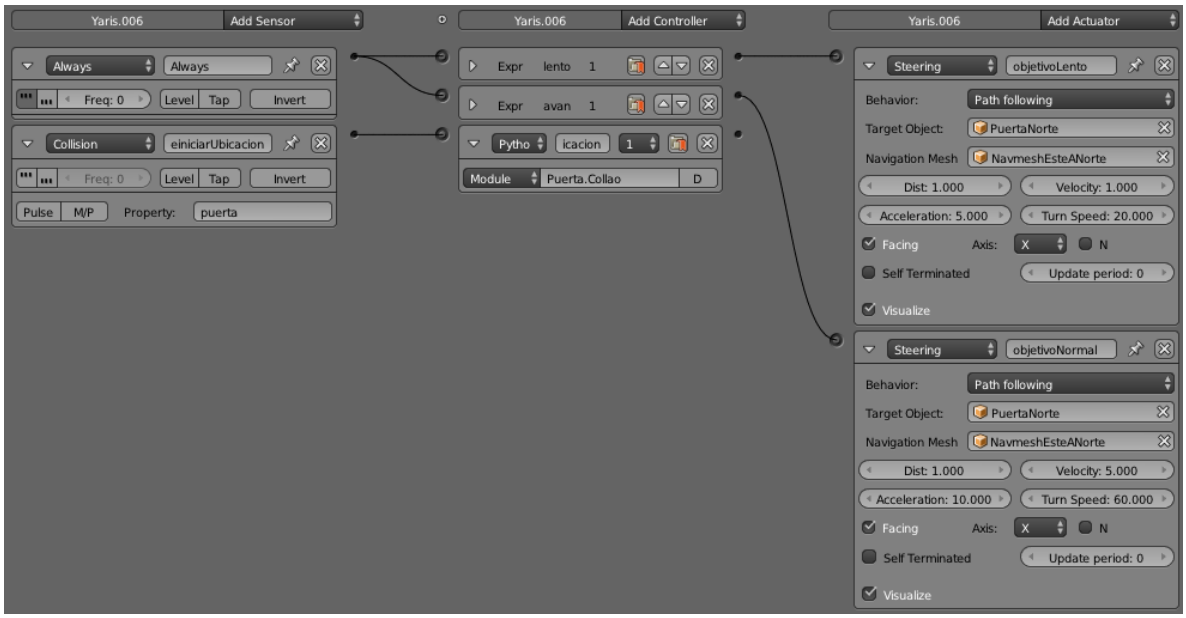


Figura lógica movimiento destino PuertaOeste

Universidad del Bío-Bío. Red de Bibliotecas - Chile

PuertaSur

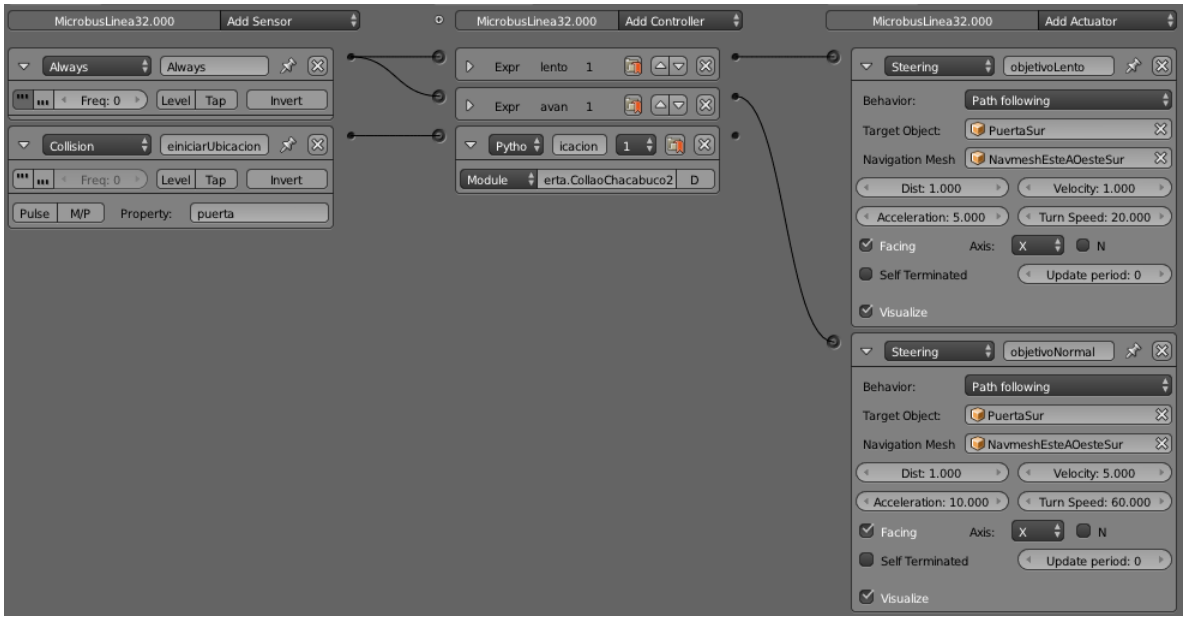


Figura lógica movimiento destino PuertaOeste

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Los vehículos que circulan desde avenida Los Carrera, pueden optar por dos destinos PuertaEste o PuertaNorte.

PuertaEste

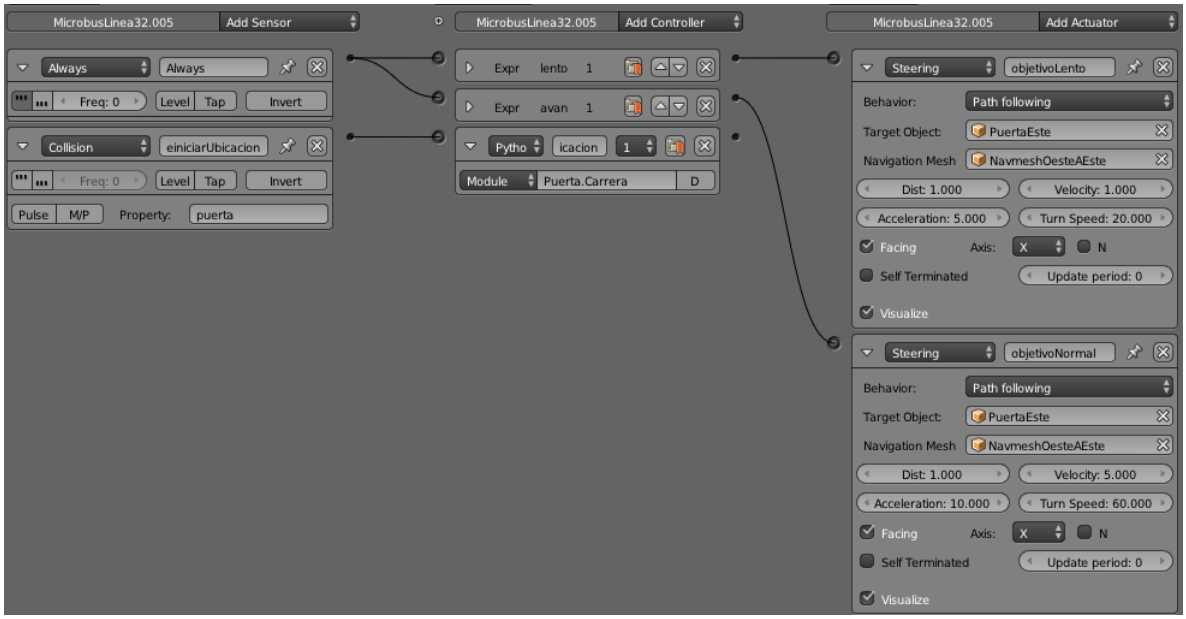


Figura lógica movimiento destino PuertaEste

PuertaNorte

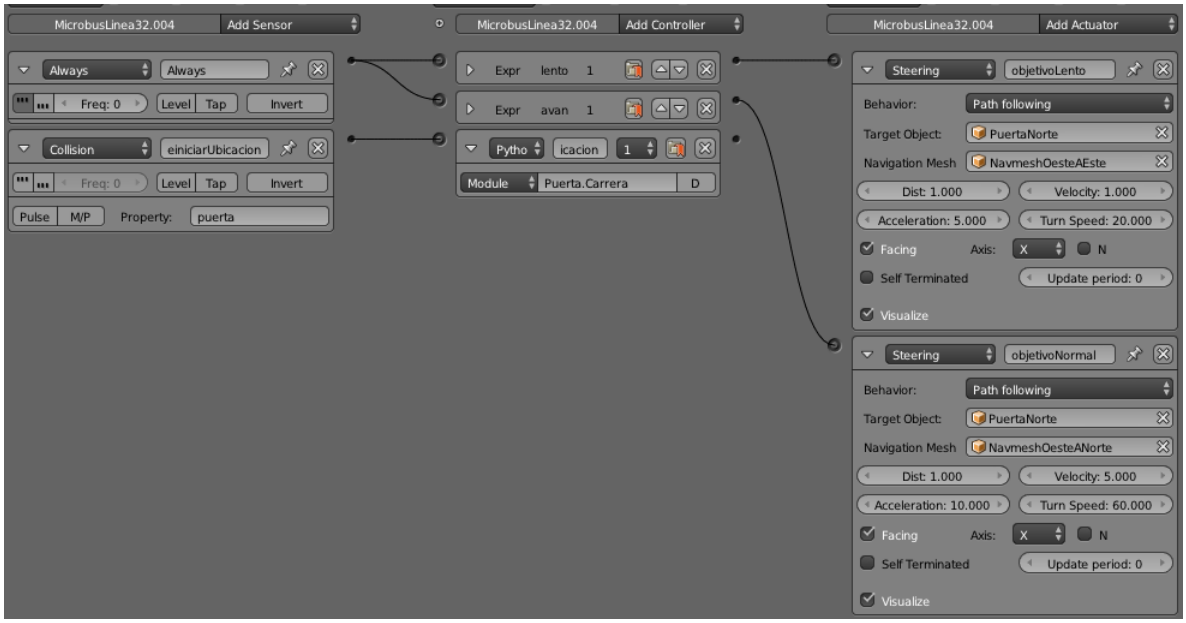


Figura lógica movimiento destino PuertaNorte

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Y los vehículos que circulan desde avenida Irarrázaval, pueden optar sólo por la PuertaNorte.

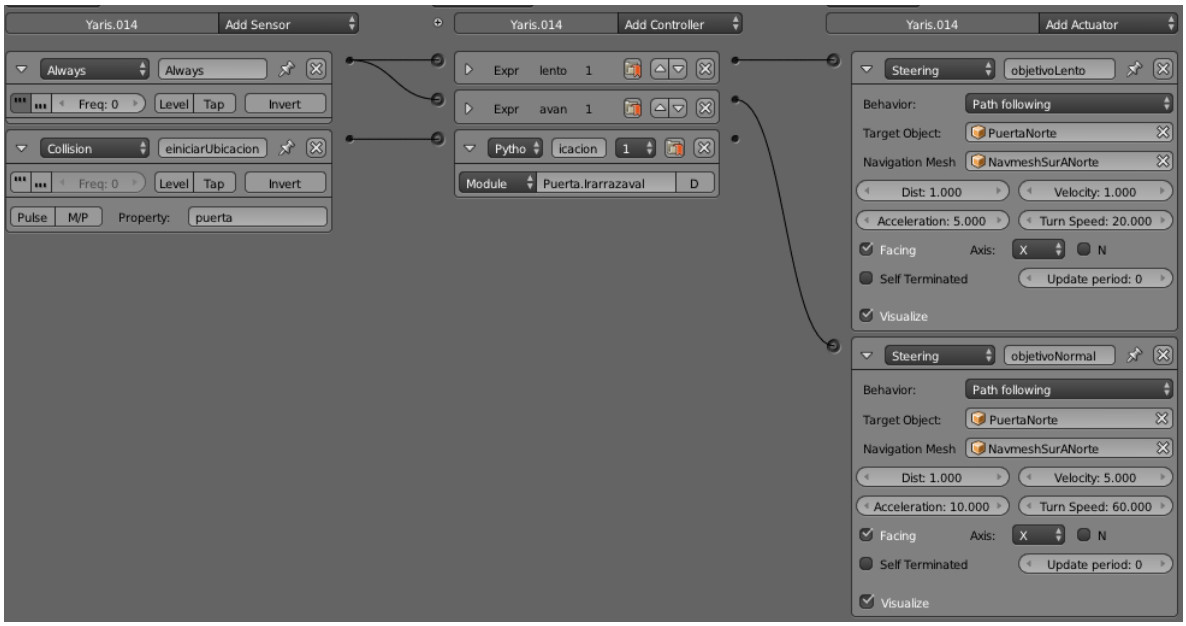


Figura lógica movimiento destino PuertaNorte

Observación respecto a la malla de navegación, en este proyecto no siempre es la misma debido a que la función path following a veces no busca el camino esperado, sino que, la ruta más corta, entonces puede darse el caso en que el vehículo circule en sentido contrario, situación que en el mundo real no es correcto.
Por ello es que en este escenario se diseñaron siete mallas de navegación en total.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.7.3. Control vehículo.

El control de vehículos se realiza mediante los actuadores de tipo motion, proporcionada por Blender.

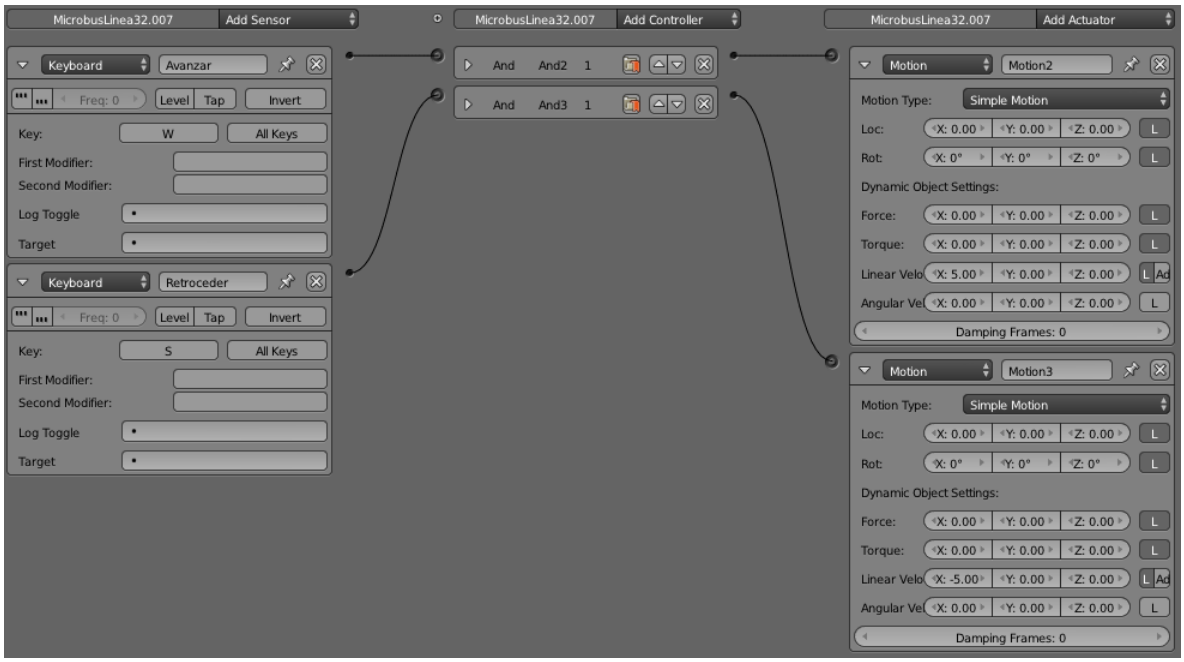


Figura lógica movimiento vehículo controlado 1.

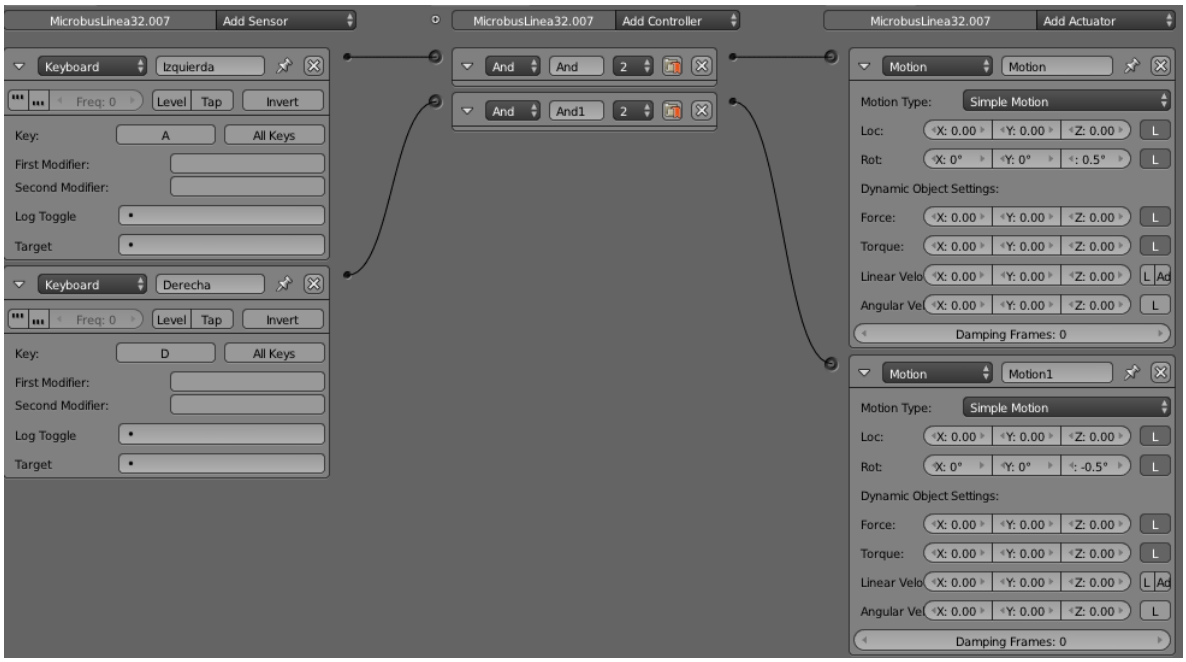


Figura lógica movimiento vehículo controlado 2.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

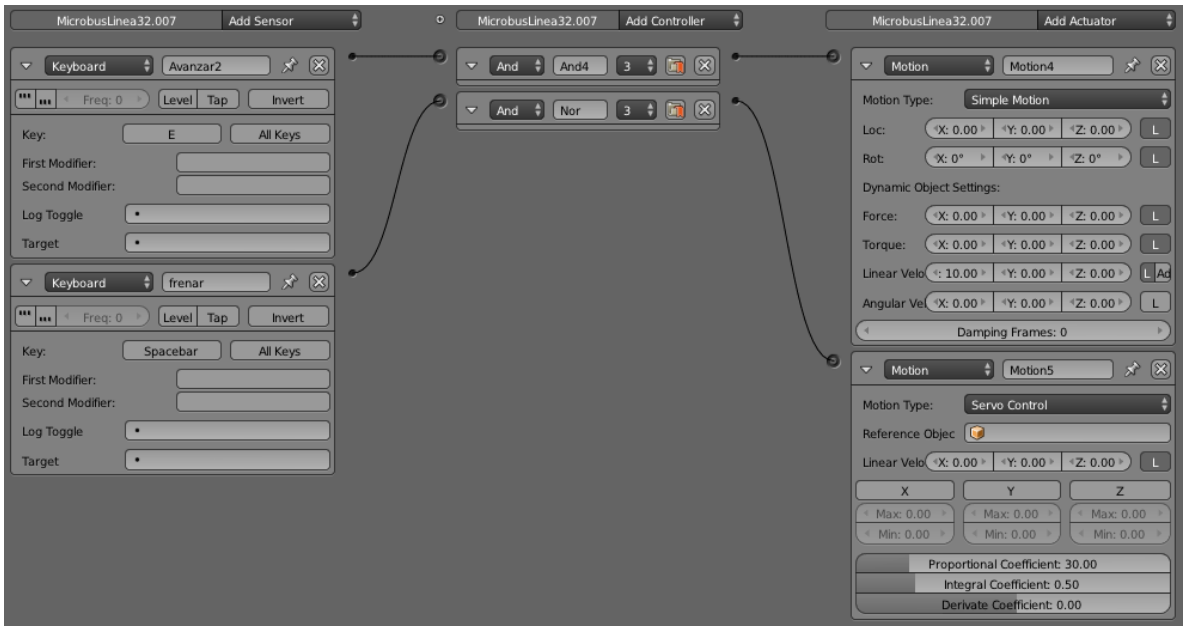


Figura lógica movimiento vehículo controlado 3.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.7.4. Movimiento de cámara.

Al igual que el control de vehículos, el movimiento de cámara se realiza mediante los actuadores de tipo motion, proporcionada por Blender, además de utilizar una tecla para activar la cámara, simulando la vista de un conductor en primera persona.

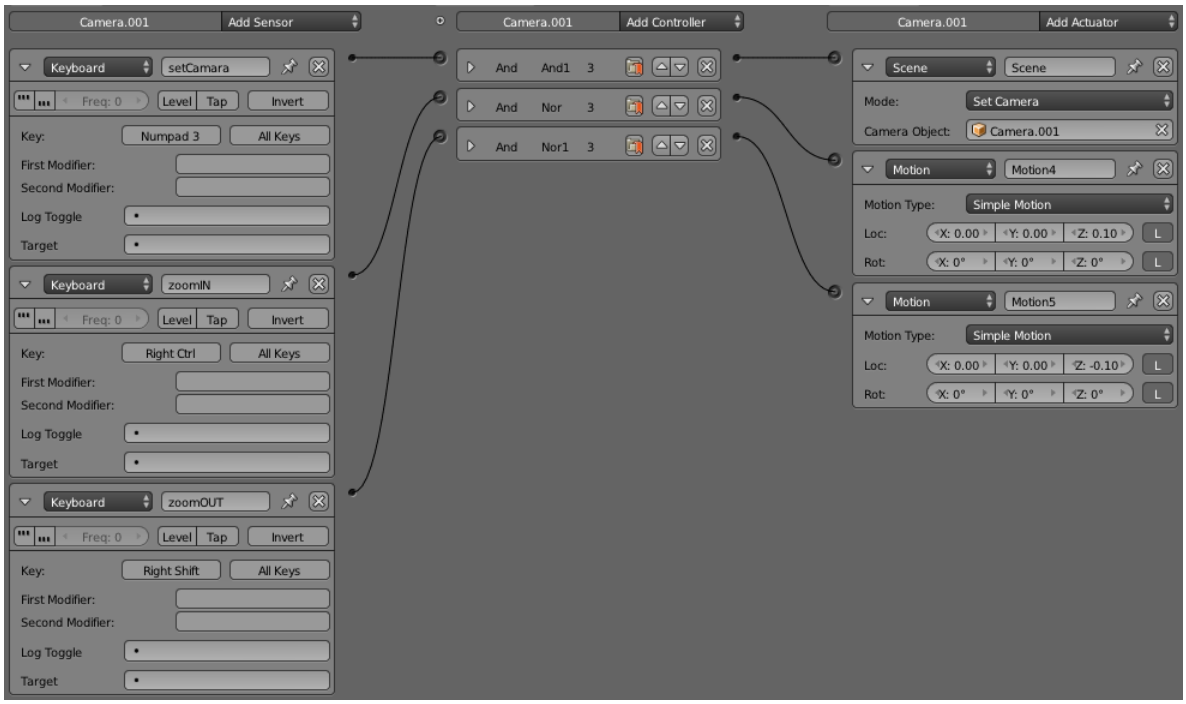


Figura lógica movimiento cámara controlada 1.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

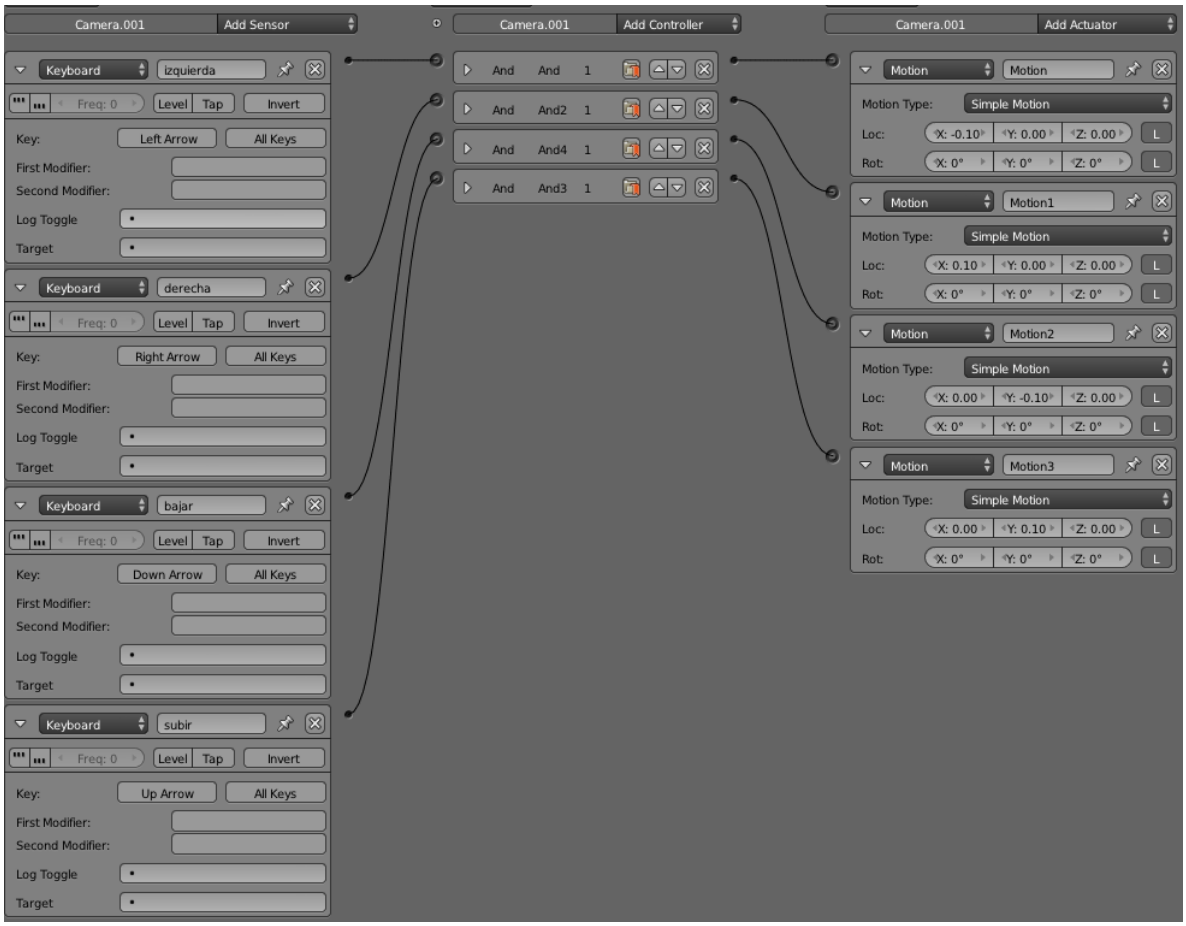


Figura lógica movimiento cámara controlada 2.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

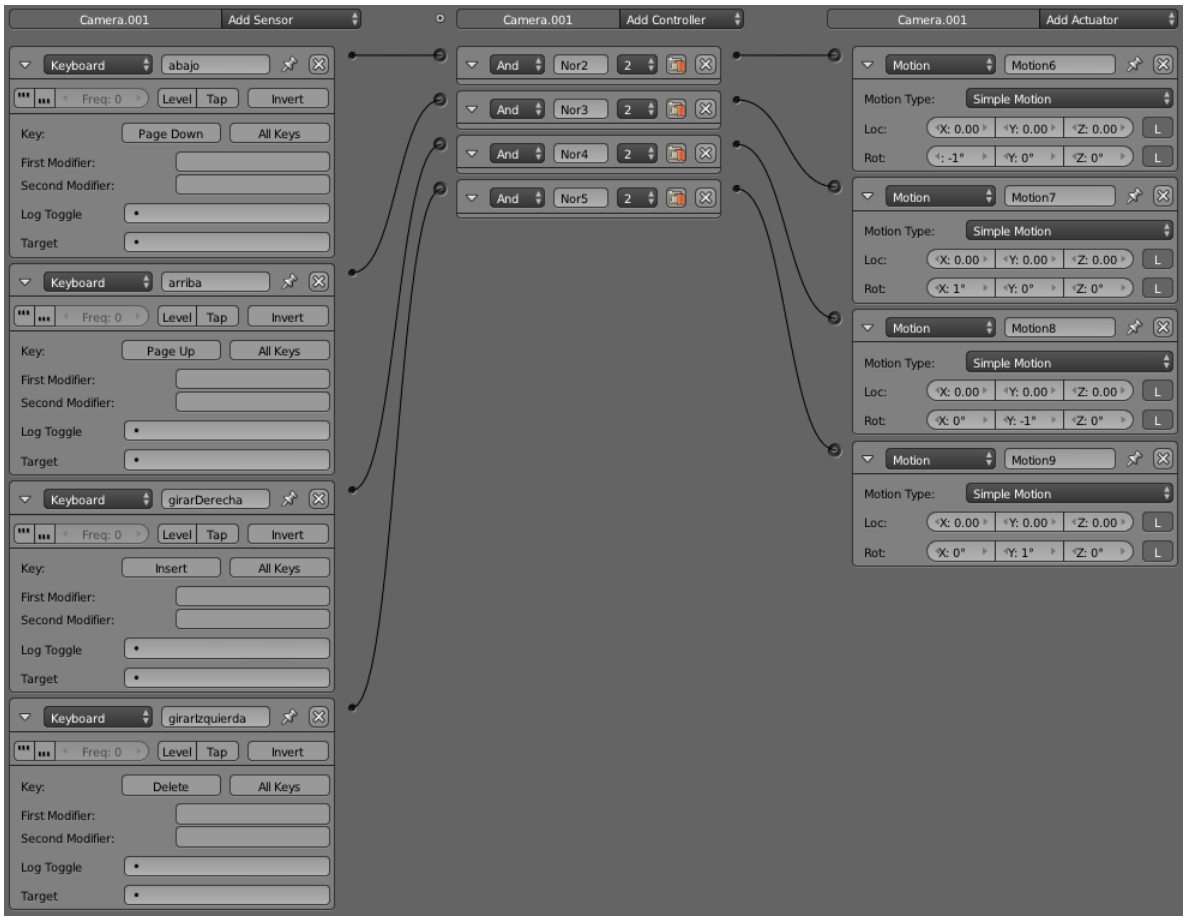


Figura lógica movimiento cámara controlada 3.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.7.5. Movimiento de peatones.

Al igual que con los vehículos, lo primero que se debe hacer es establecer el objetivo o meta de cada peatón, para esto se fijan las puertas como objetivos en este caso quedando estipuladas así:

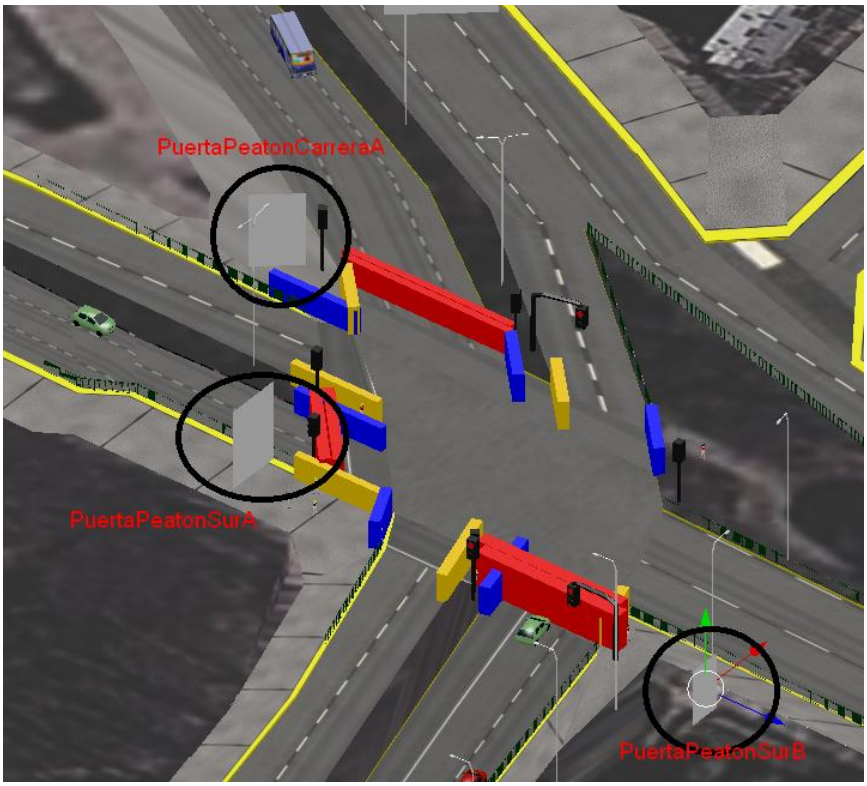


Figura ubicación puertas peatones.

Así, los peatones que circulan por el escenario pueden optar por 3 destinos PuertaPeatonCarreraA, PuertaPeatonSurA o PuertaPeatonSurB.
Para fijar el destino apropiado se debe establecer cambiando la propiedad Target Object, fijando el objeto adecuado, además de fijar la malla de navegación correspondiente a los peatones de la siguiente forma:

Universidad del Bío-Bío. Red de Bibliotecas - Chile

PuertaPeatonCarreraA

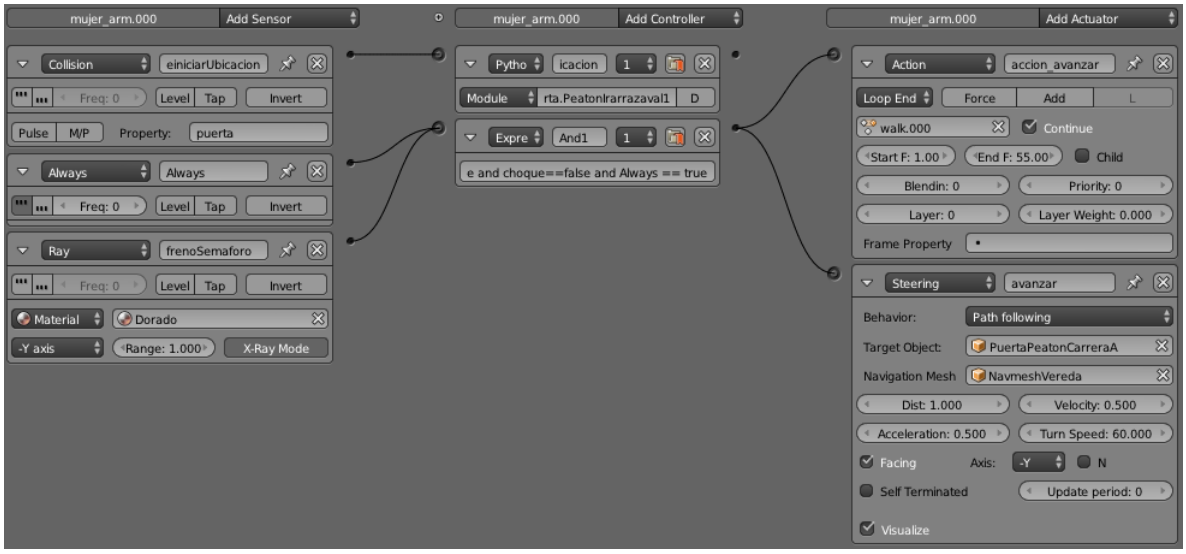


Figura lógica movimiento destino PuertaPeatonCarreraA.

PuertaPeatonSurA

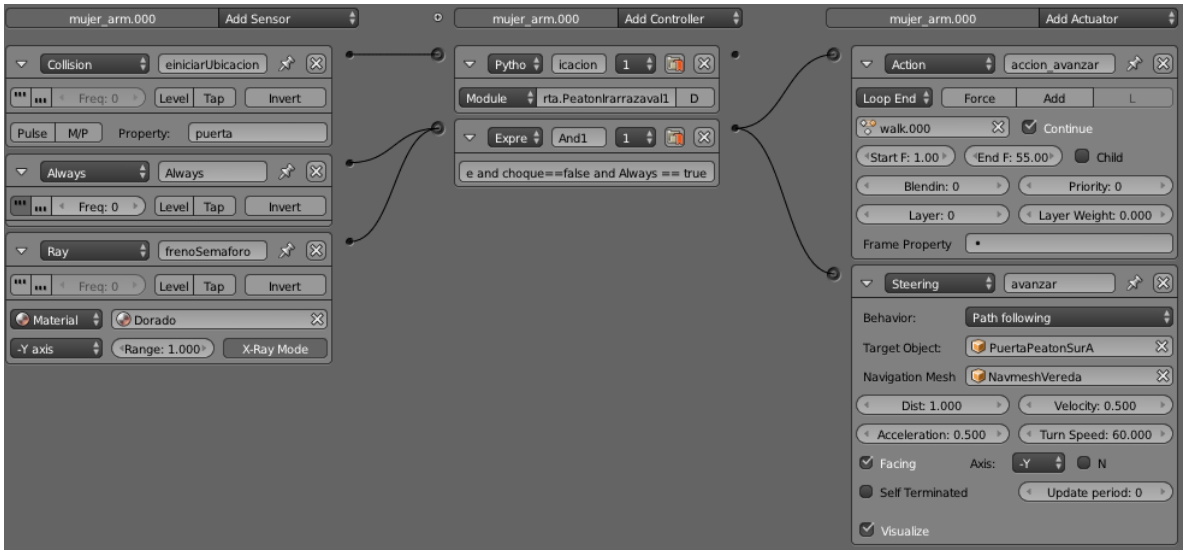


Figura lógica movimiento destino PuertaPeatonSurA.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

PuertaPeatonSurB

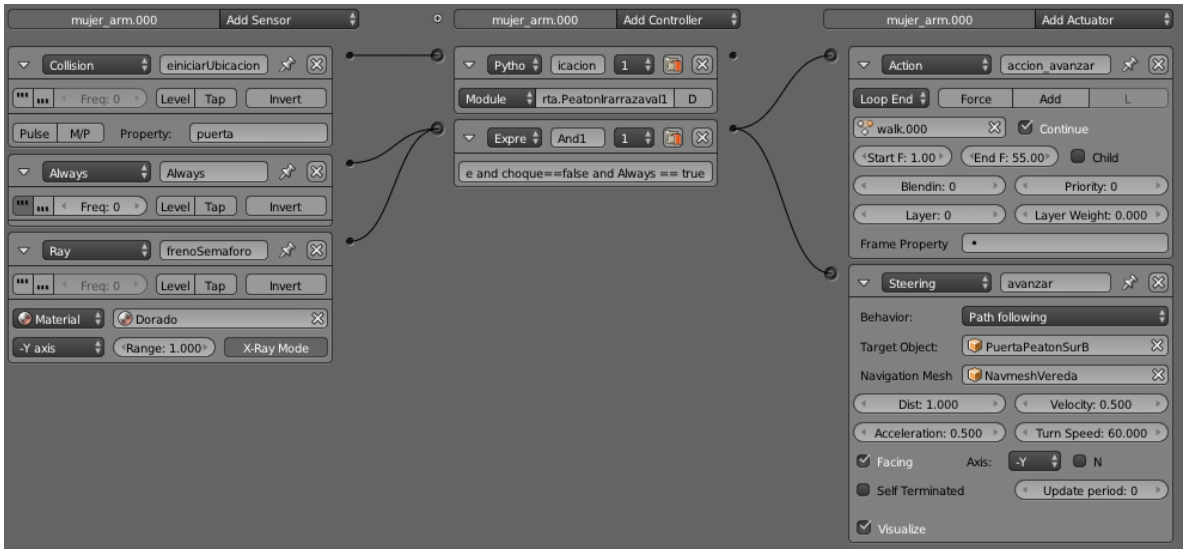


Figura lógica movimiento destino PuertaPeatonSurB.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.7.6. Sincronización semáforos

Como se presentó anteriormente, para la simulación de semáforos se utilizan dos objetos, uno que sirve para ser detectado ya sea por vehículos o peatones dependiendo del caso, pero no es visible durante la ejecución y otro que sirve sólo para ser visible que representa la simulación de un semáforo para el observador.

7.7.6.1. Semáforos Avenida Collao.

Semáforos funcionales Avenida Collao.

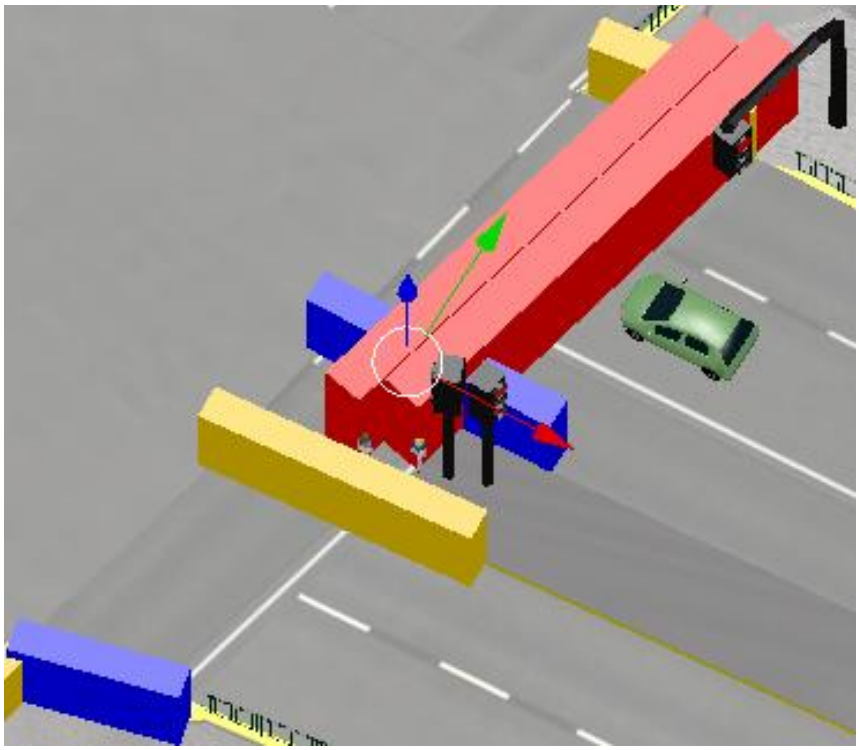


Figura Barreras semáforos avda. Collao.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

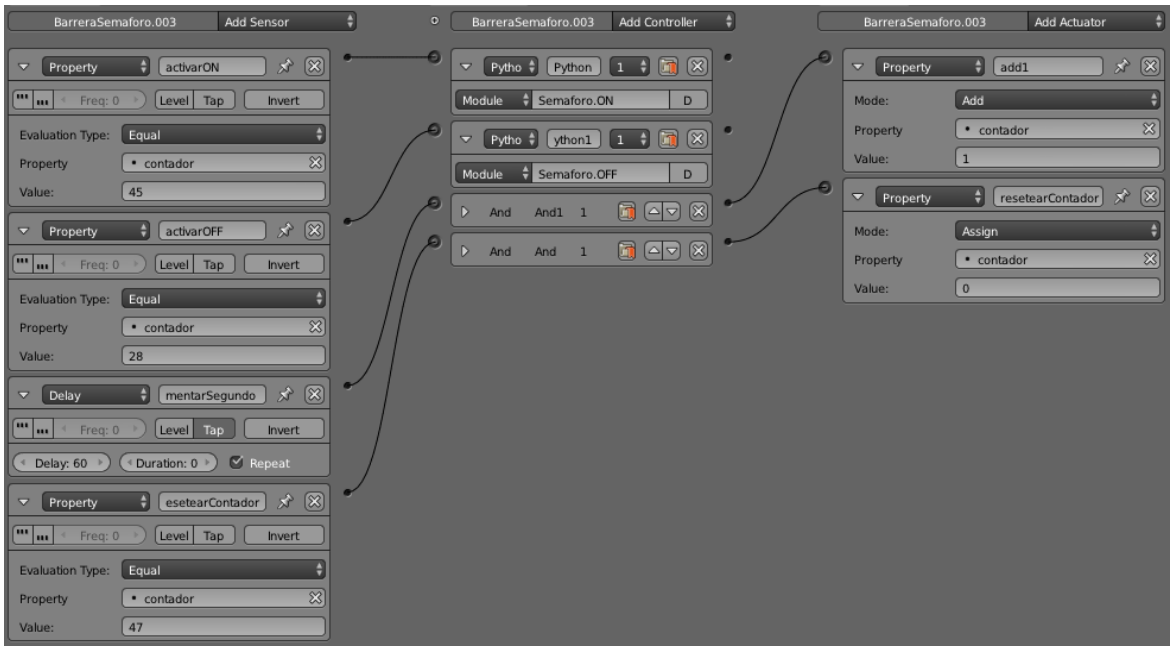


Figura lógica barrera semáforos vehículos avda. Collao.

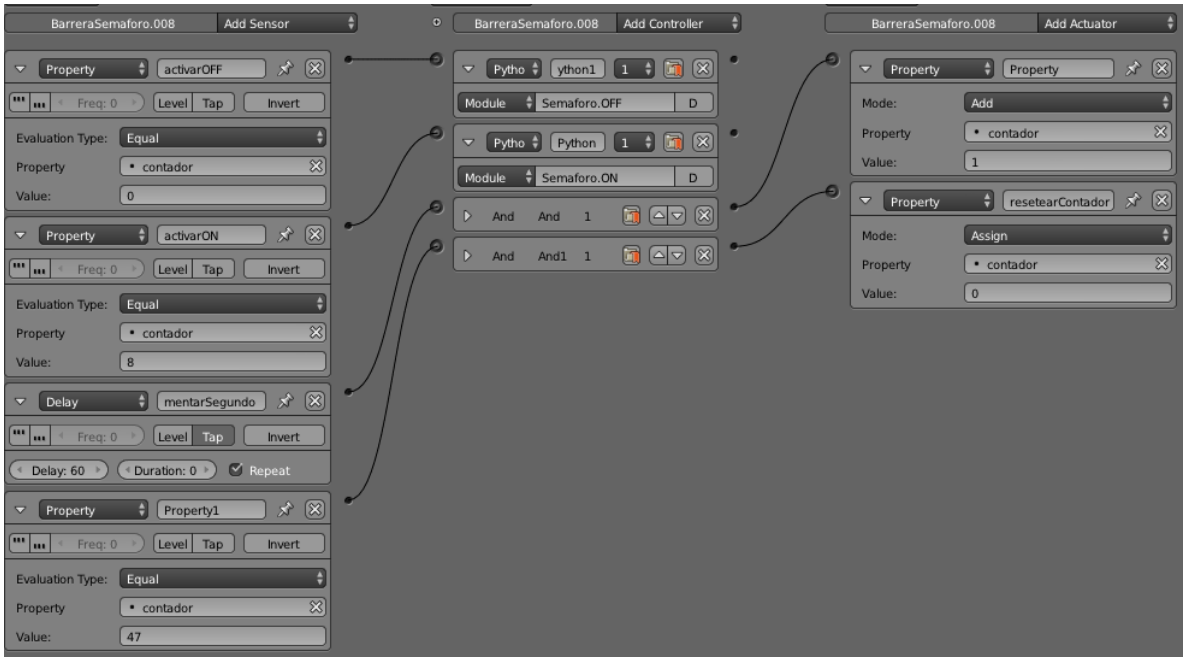


Figura lógica barrera semáforos peatones avda. Collao.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Semáforo Visual Avenida Collao

Lógica color verde

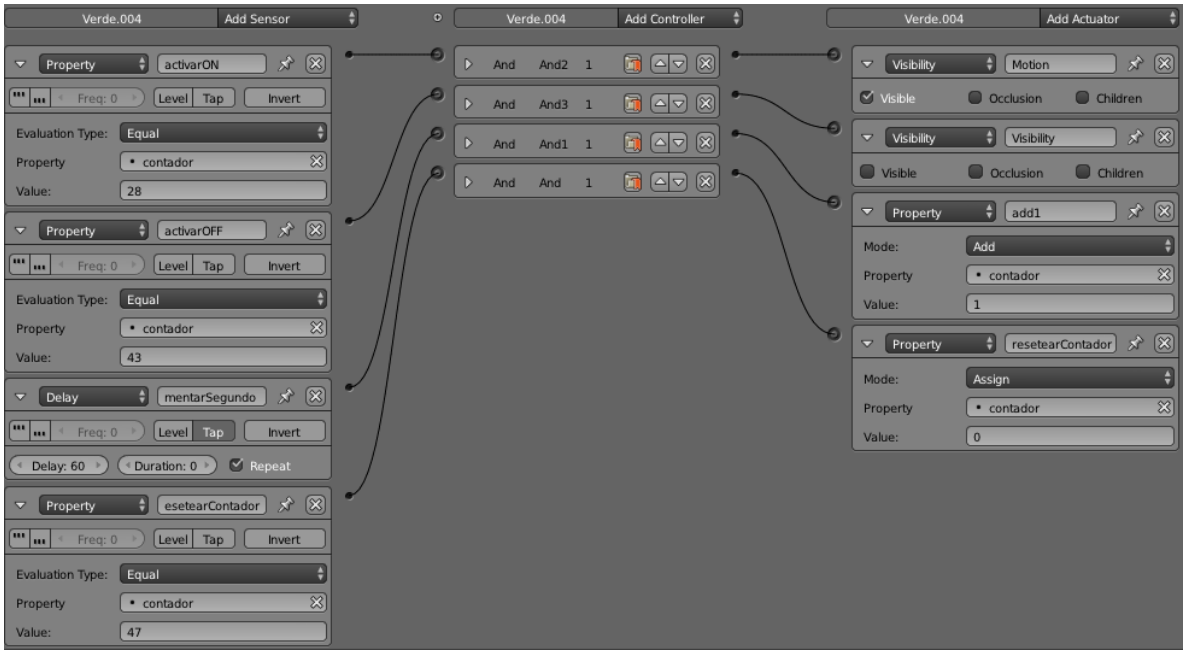


Figura lógica semáforo vehículos color verde avda. Collao.

Lógica color amarillo

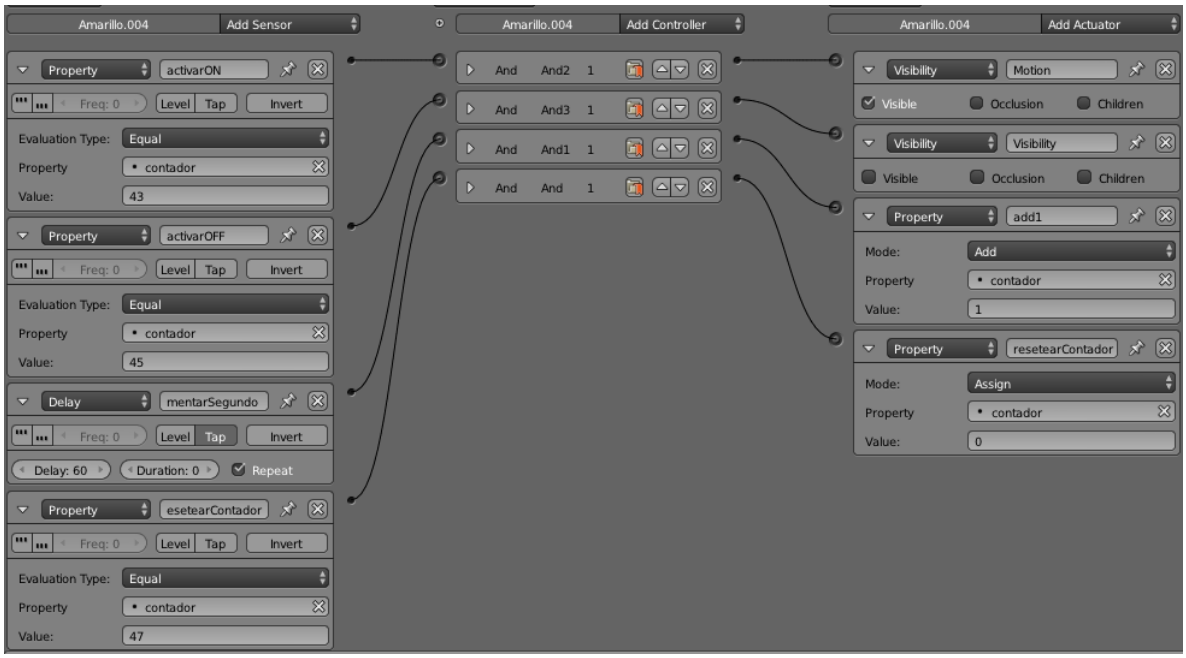


Figura lógica semáforo vehículos color amarillo avda. Collao.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica color rojo

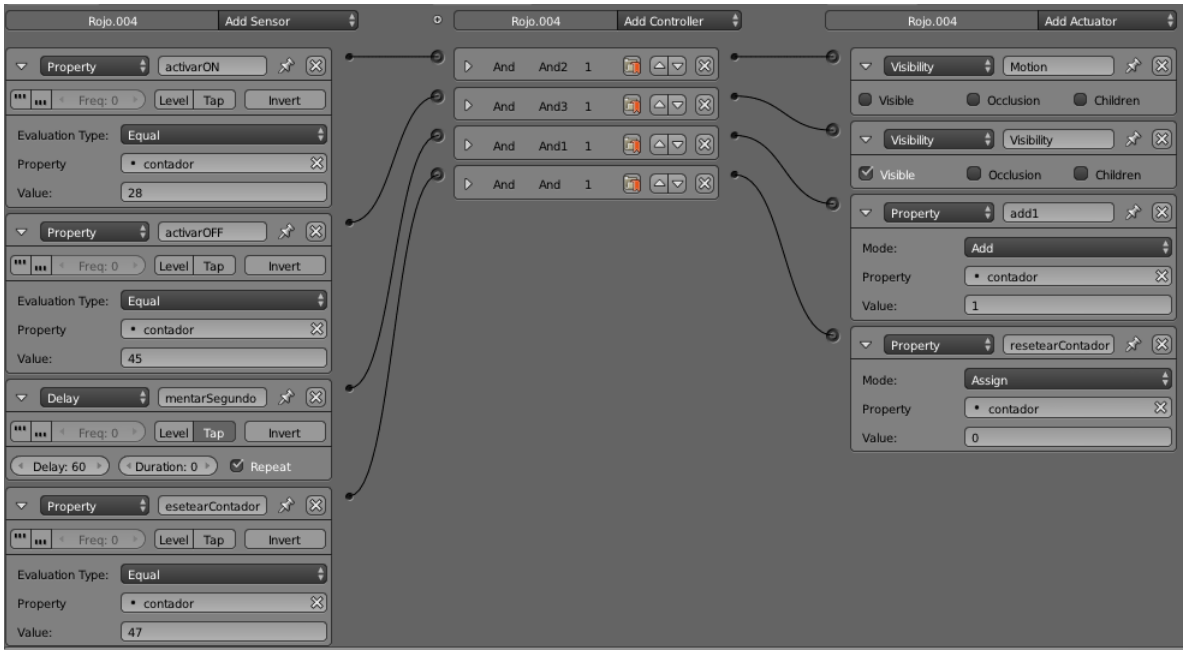


Figura lógica semáforo vehículos color rojo avda. Collao.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.7.6.2. Semáforos Avenida Irarrázaval.

Semáforos funcionales Avenida Irarrázaval.

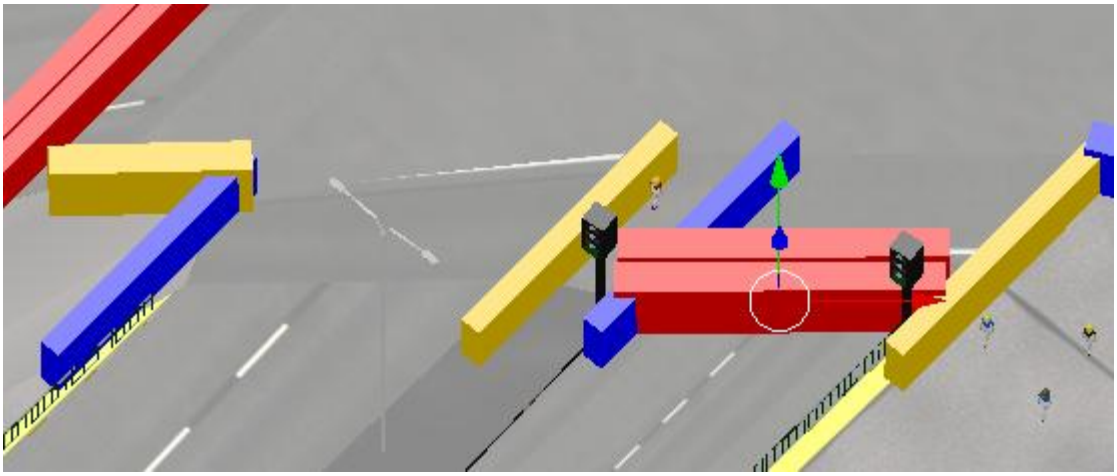


Figura Barreras semáforos avda. Irarrázaval.

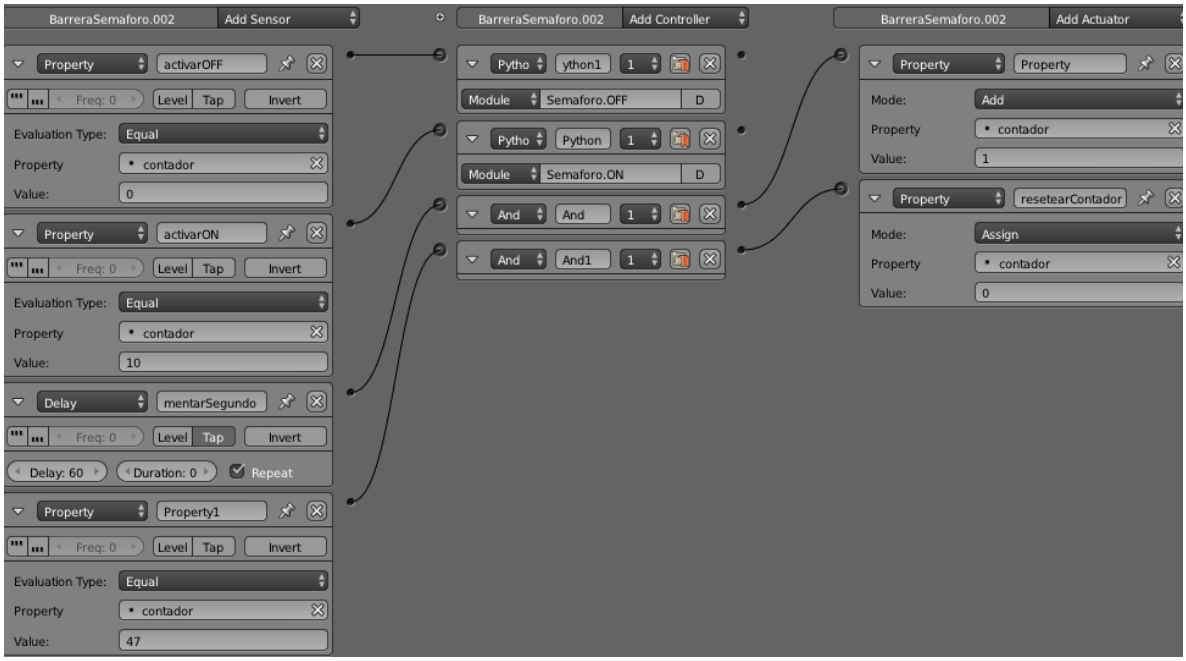


Figura lógica barrera semáforos vehículos avda. Irarrázaval.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

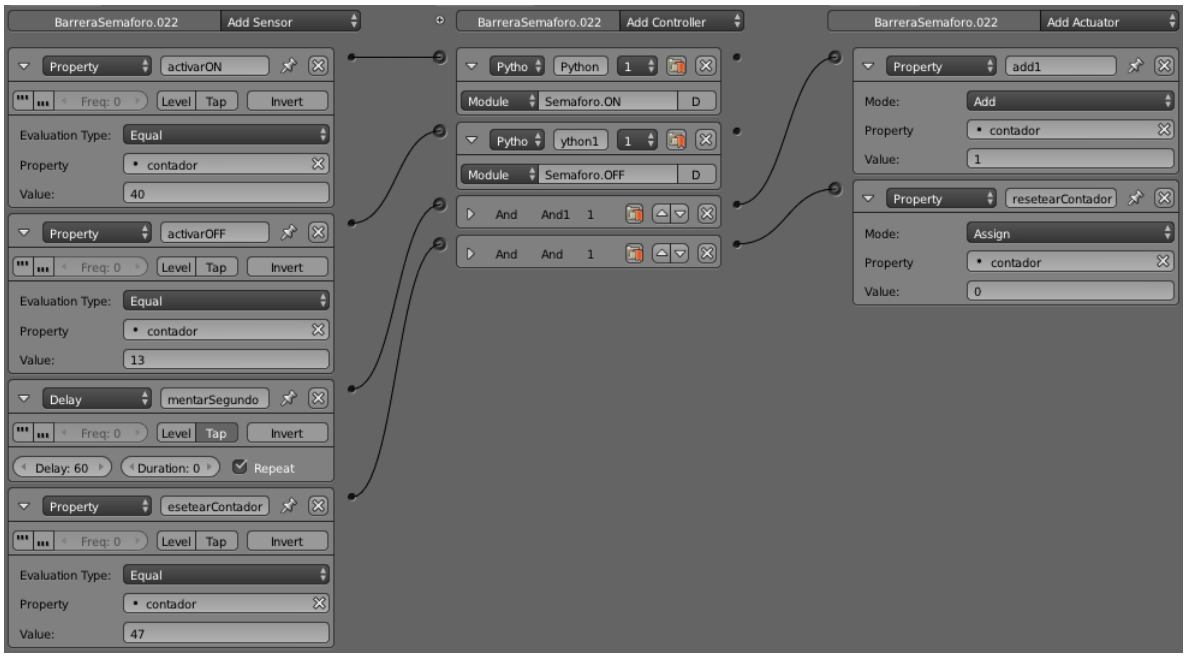


Figura Barreras semáforos peatones avda. Irarrázaval.

Semáforo Visual Avenida Irarrázaval

Lógica color verde

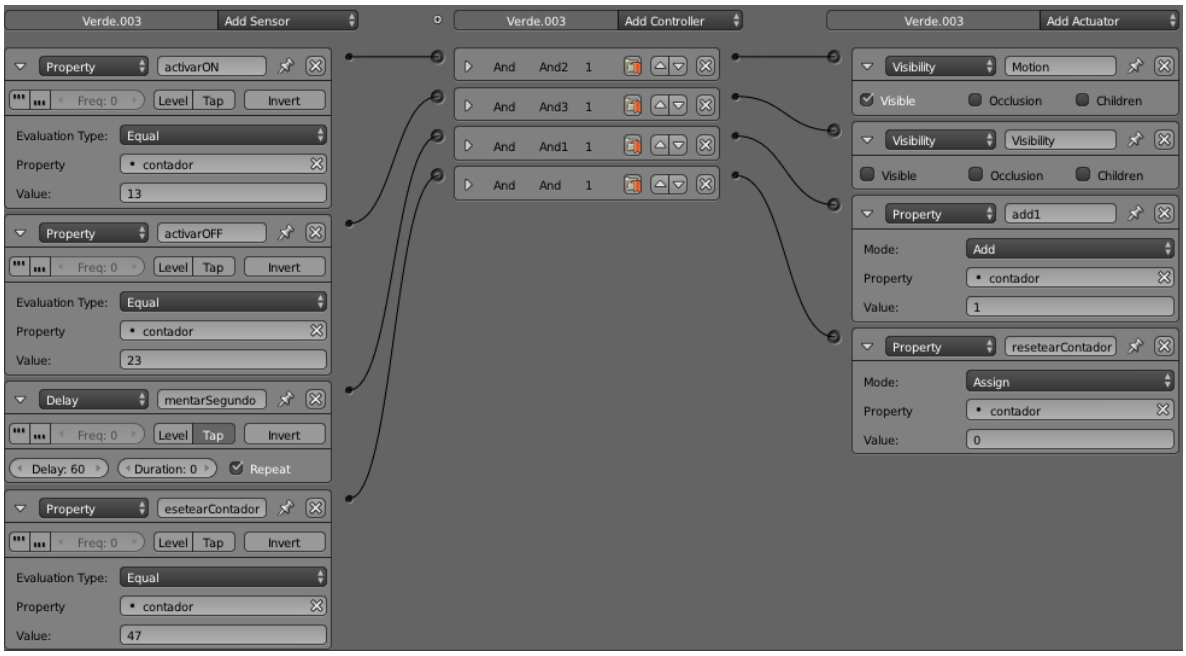


Figura lógica semáforo vehículos color verde avda. Irarrázaval.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica color amarillo

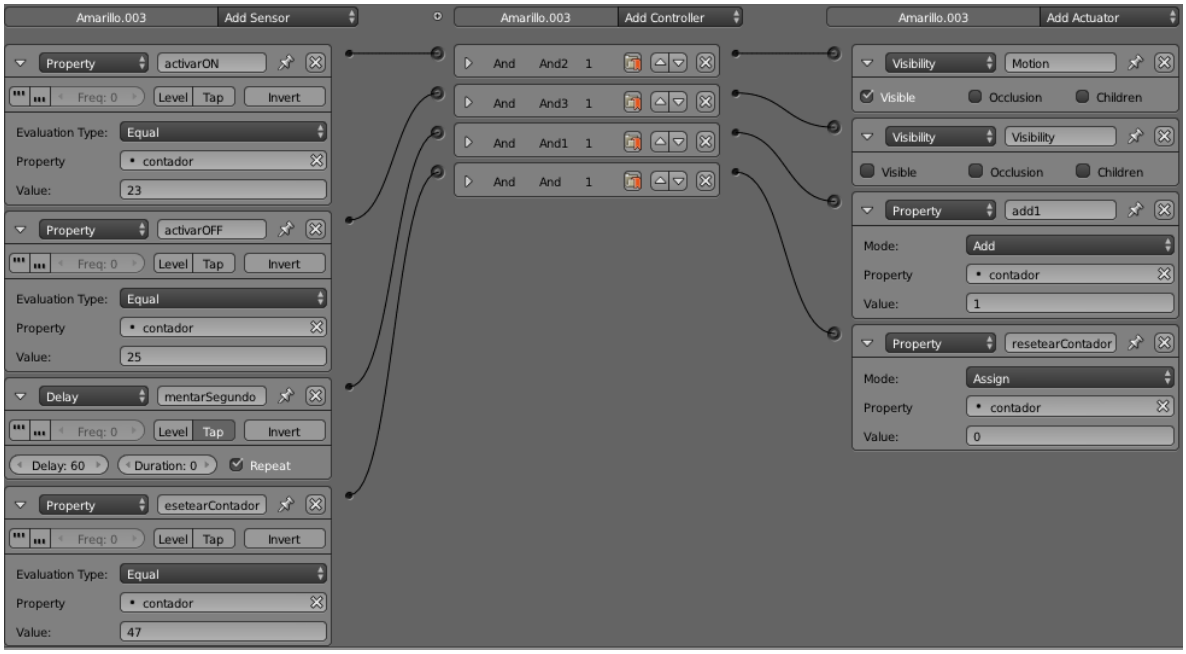


Figura lógica semáforo vehículos color amarillo avda. Irarrázaval

Lógica color rojo

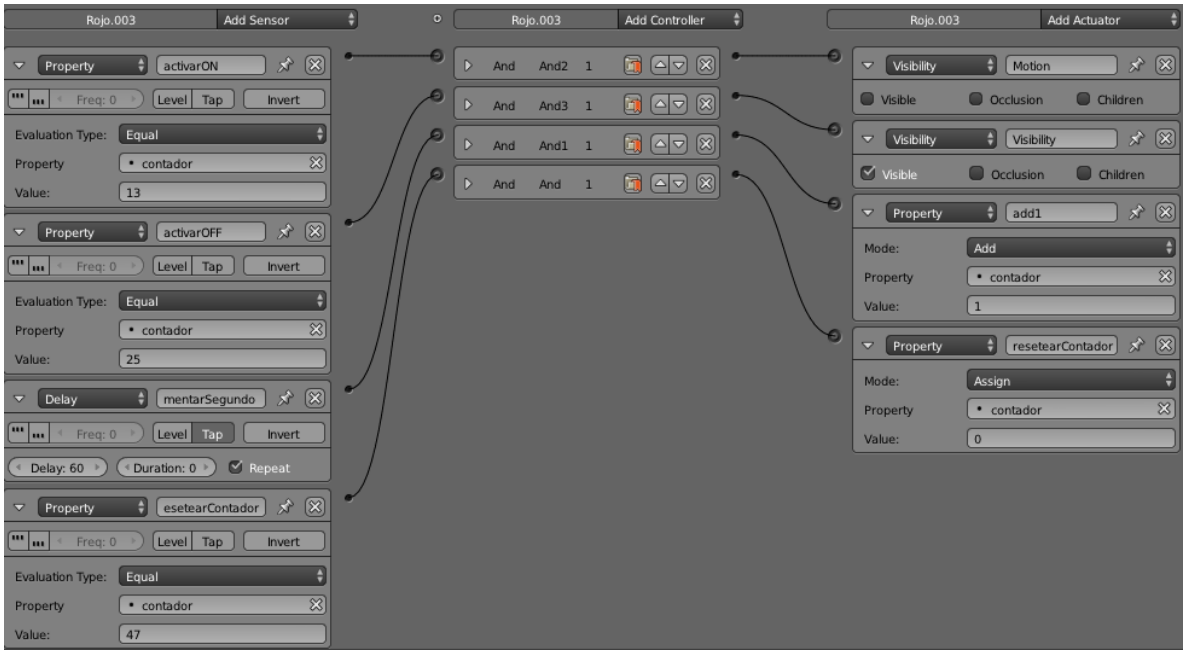


Figura lógica semáforo vehículos color rojo avda. Irarrázaval.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.7.6.3. Semáforos Avenida Los Carrera.

Semáforos funcionales Avenida Los Carrera.

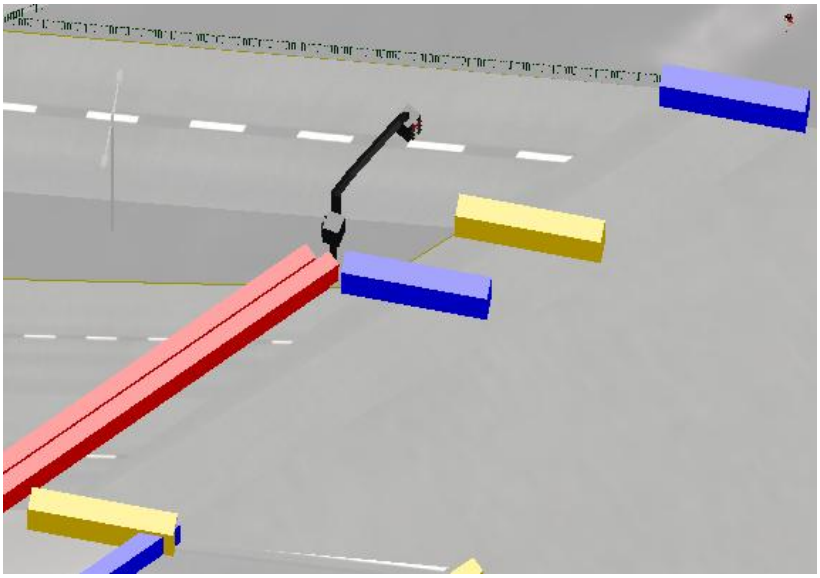


Figura Barreras semáforos avda. Los Carrera.

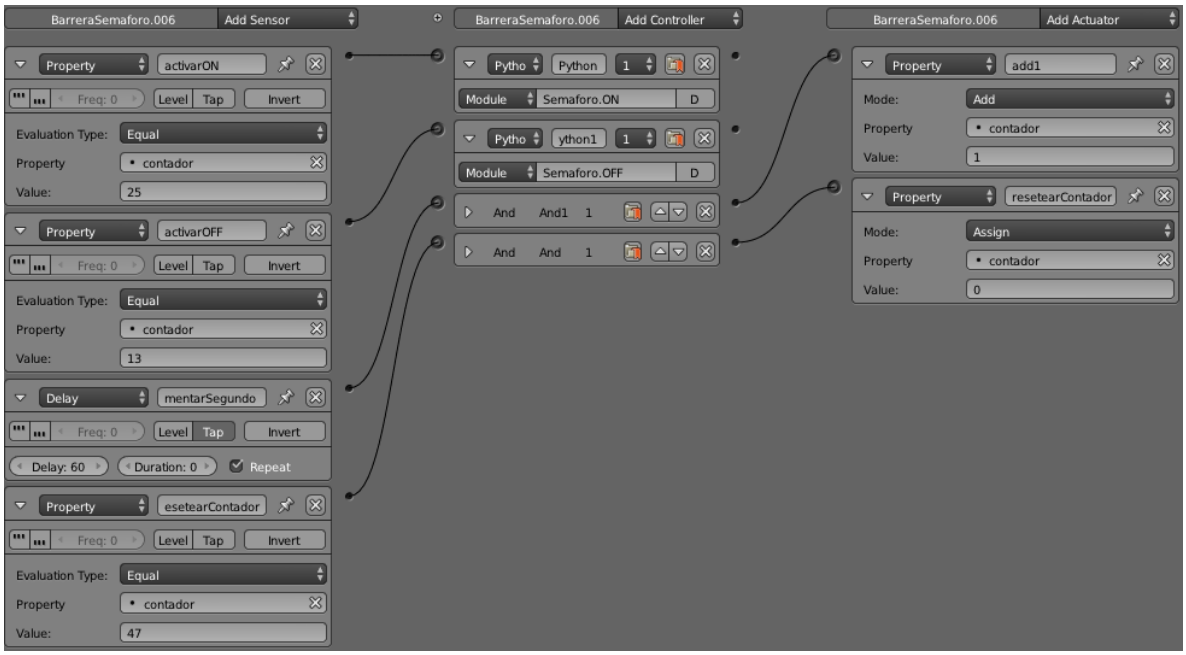


Figura lógica barrera semáforos vehículos avda. Los Carrera.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

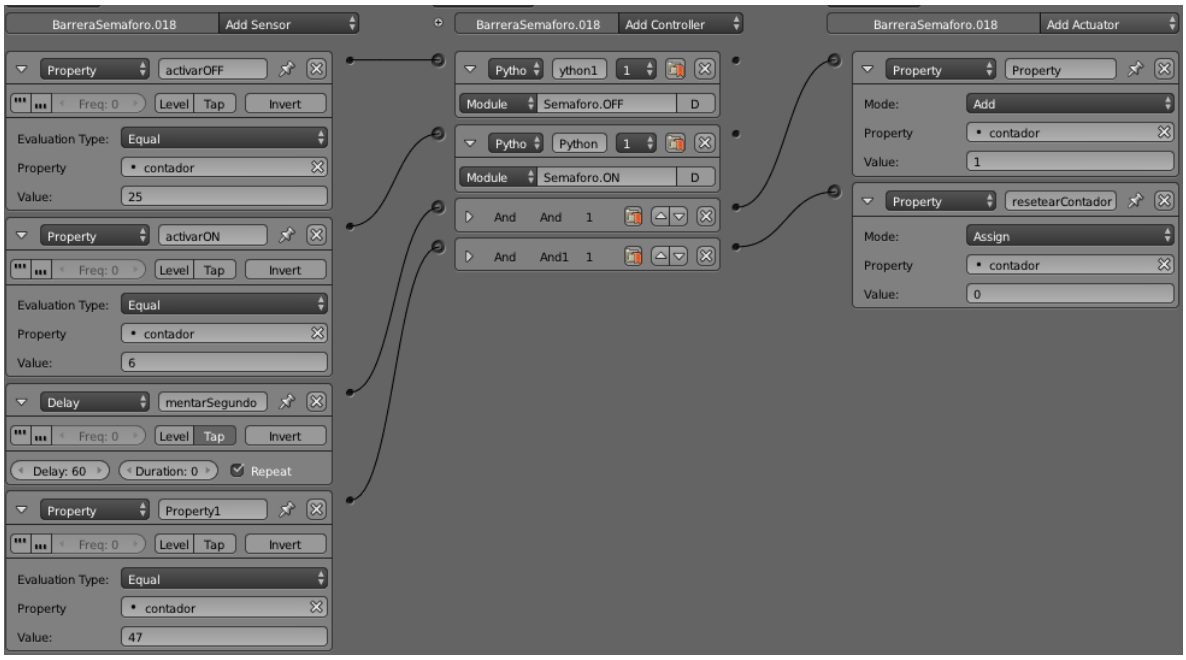


Figura lógica barrera semáforos peatones avda. Los Carrera.

Semáforo Visual Avenida Los Carrera

Lógica color verde

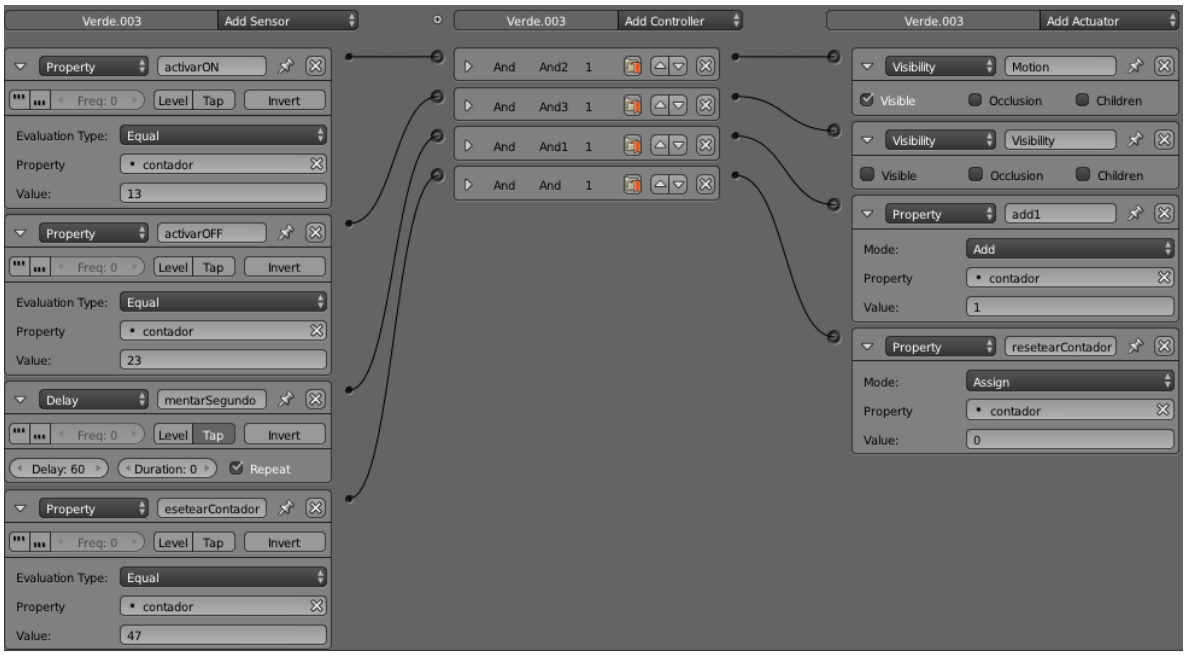


Figura lógica semáforo vehículos color verde avda. Los Carrera.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Lógica color amarillo

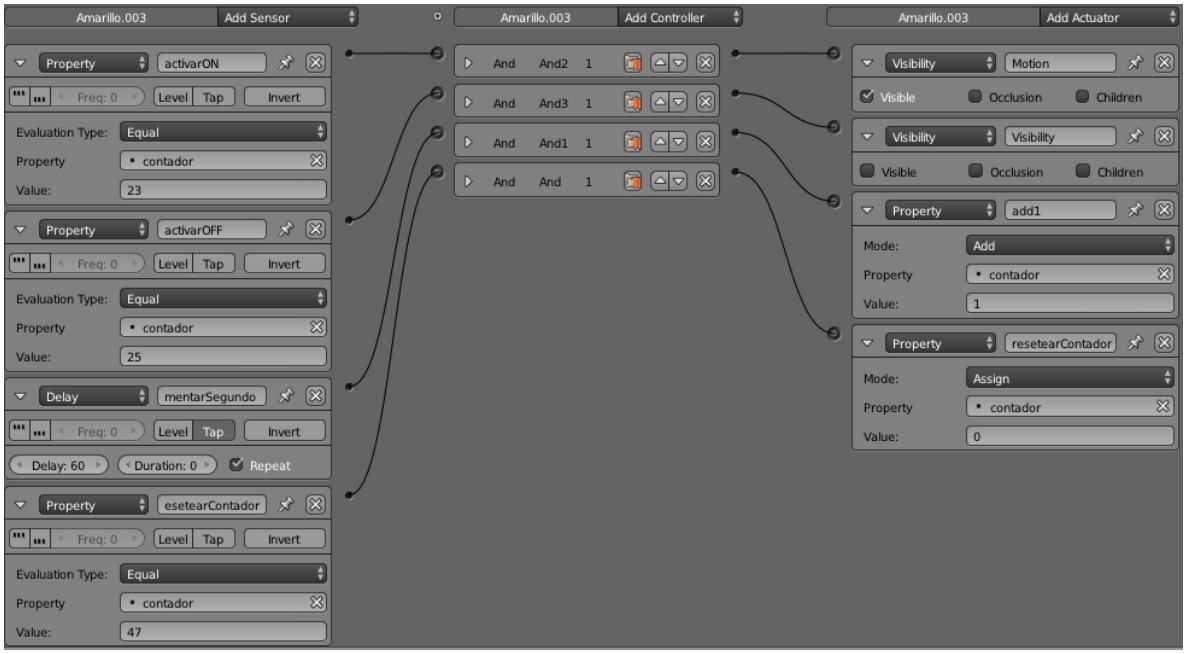


Figura lógica semáforo vehículos color amarillo avda. Los Carrera.

Lógica color rojo

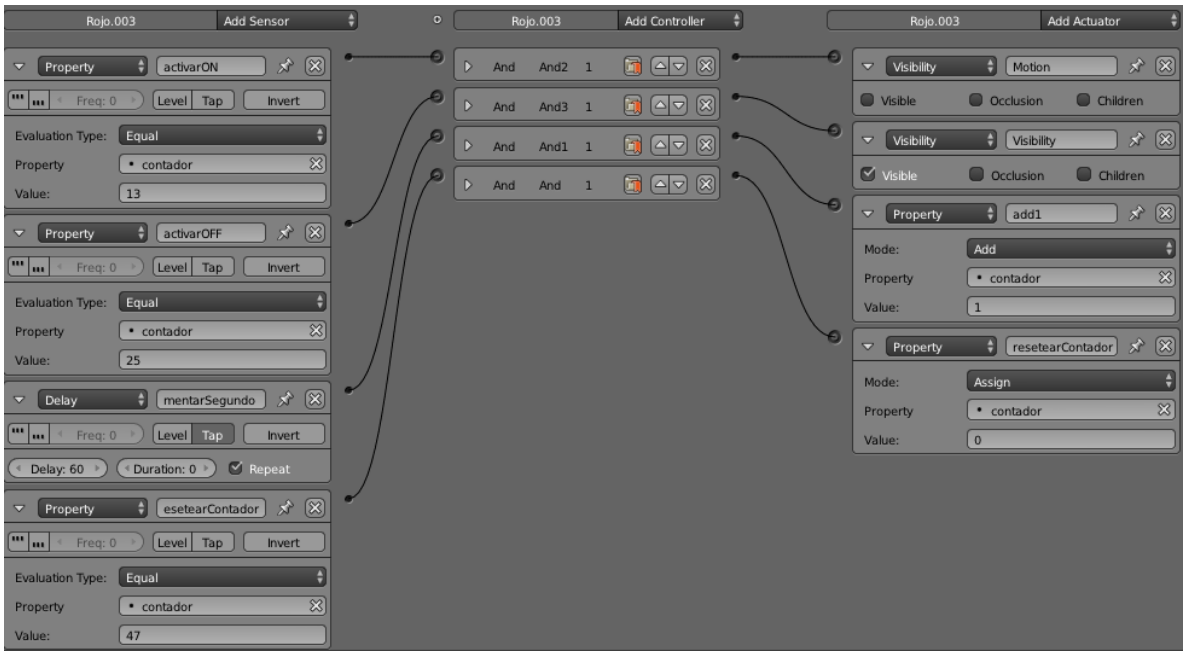


Figura lógica semáforo vehículos color rojo avda. Los Carrera.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Observación

Respecto a los colores con los cuales se representan las barreras semáforo de peatones, éstas poseen la misma lógica independiente del color de material, el color es utilizado para no detener a los peatones en medio de la calzada, como se dijo en la sección 7.6.2 Componentes de humanos.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

7.8. Validar diseño final.

En esta etapa se evaluó el cumplimiento de los objetivos iniciales del escenario final, por parte de los encargados del proyecto.

El proyecto cumplió con la compatibilidad existente debido a que es utilizado Blender en el desarrollo, ya sea de los objetos y del escenario simulado.

Dicho escenario permite la simulación de peatones y vehículos además cumple con el control de un vehículo el cual puede moverse de según lo indique el usuario.

Finalmente el escenario virtual desarrollado, cumple con los objetivos del proyecto, simula correctamente el tránsito de peatones y vehículos.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

8. LA APLICACIÓN.

Para poder ejecutar el escenario es necesario tener instalado Blender, abrir el archivo correspondiente al escenario y luego presionar la tecla "P" para iniciarlo y la tecla "Escape" para terminarlo

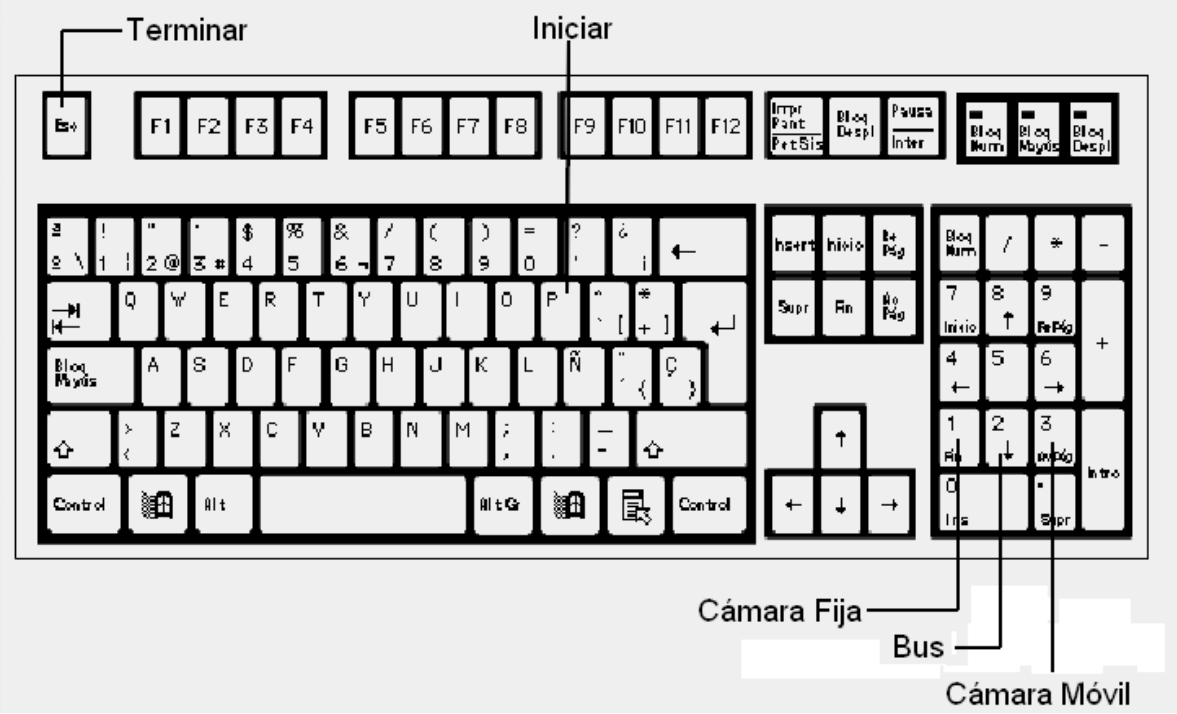


Figura controles generales.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Controles cámara móvil

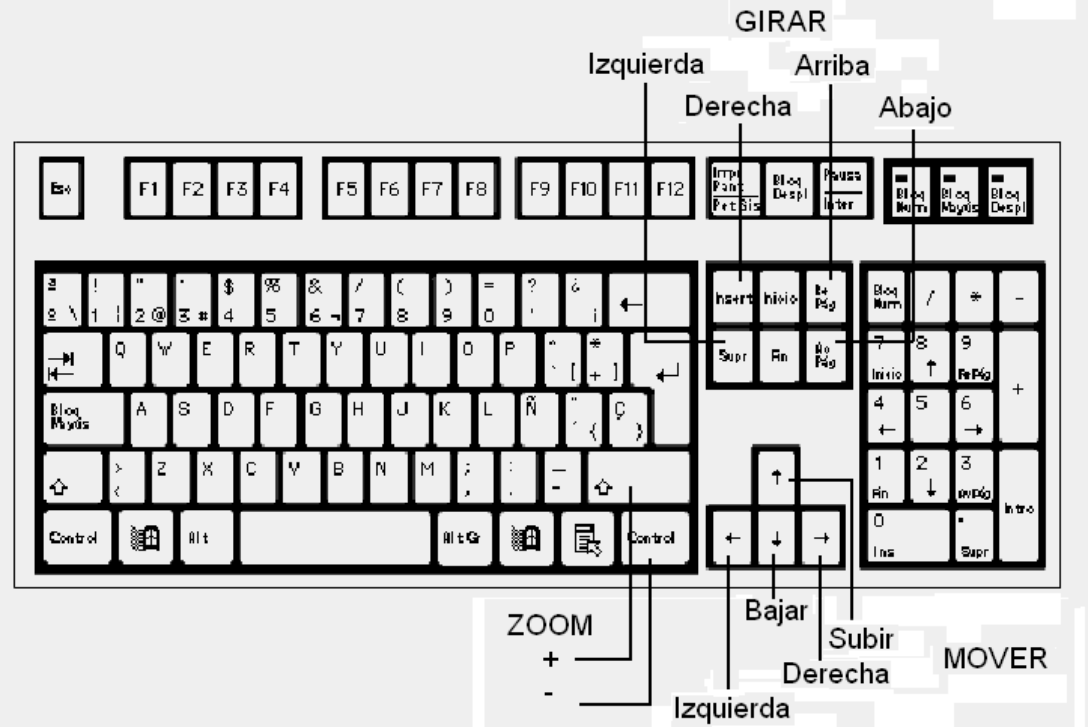


Figura controles cámara móvil.

Controles Vehículo

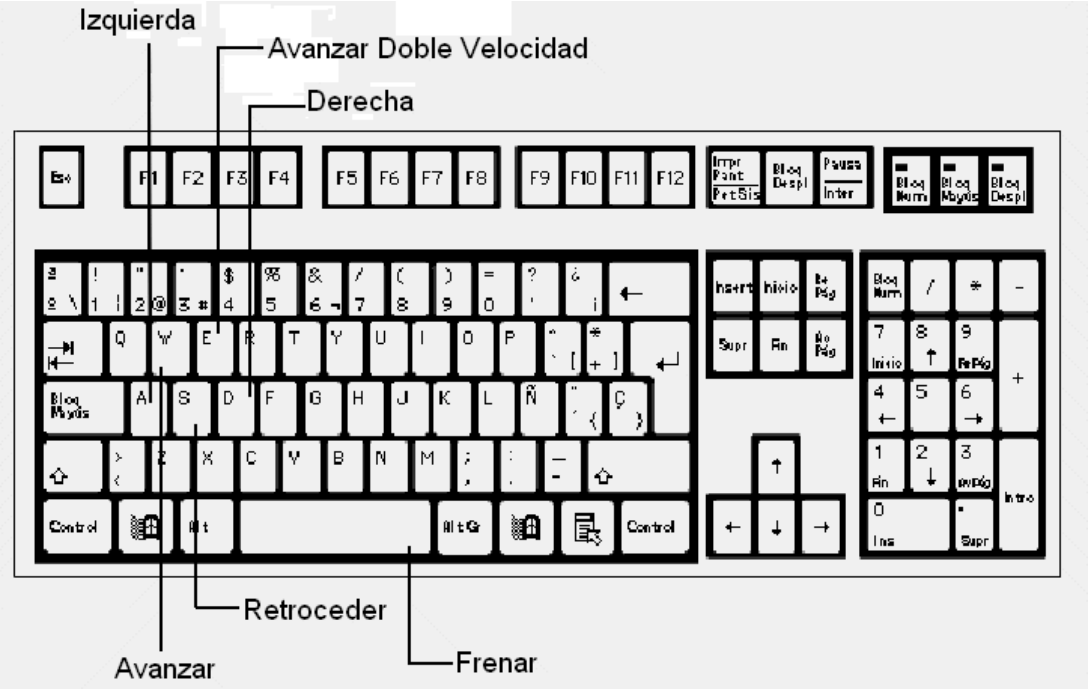


Figura controles vehículo.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

8.1. Tabla Descriptiva de objetos

La siguiente tabla detalla los objetos utilizados y sus principales componentes

Vehículos:

Microbús:		
Propiedades		
	frenar	Se activa cuando es preestablecido frenar o disminuir la velocidad. Ejemplo: cerca de un objeto tipo vehículo.
	vehículo	Propiedad que sirve para conocer el tipo del objeto. Ejemplo autos tipo "vehículo".
	choque	Se activa cuando se considera que hay choque. Ejemplo: colisión con un objeto tipo "vehículo".
	parar	Se activa cuando es preestablecido detenerse. Ejemplo: muy cerca de un objeto tipo "vehículo".
Sensores		
	pararRojo	Detecta la presencia de un objeto con material "rojo", frente a él.
	stopVehiculo	Detecta la presencia de un objeto con propiedad "vehículo", frente a él.
	pararPersona	Detecta la presencia de un objeto con propiedad "persona", frente a él.
	frenoProximidadPersona	Detecta la presencia de un objeto con propiedad "persona", frente a él.
	frenoProximidadVehiculo	Detecta la presencia de un objeto con propiedad "vehículo", frente a él.
	sensorSemaforoRojo	Detecta la presencia de un objeto con material "rojo", frente a él.
	choqueBarreraVerde	Detecta la colisión con un objeto de material "rojo".
	choqueVehiculo	Detecta la colisión con un objeto de propiedad "vehículo".
	choquePersona	Detecta la colisión con un objeto de propiedad "persona".
	Always	Se activa al iniciar el programa.
	reiniciarUbicacion	Se activa al llegar al objetivo, la idea es volver a una ubicación deseada.
Actuadores		
	setStopTrue	Establece la propiedad "parar" en "verdadero".

Universidad del Bío-Bío. Red de Bibliotecas - Chile

	setStopFalse	Establece la propiedad "parar" en "falso".
	setFrenar	Establece la propiedad "frenar" en "verdadero"
	setFrenarFalse	Establece la propiedad "parar" en "falso".
	setChoque	Establece la propiedad "choque" en "verdadero"
	objetivoLento	Direcciona el objeto a otro especificado a una velocidad menor a la normal.
	objetivoNormal	Direcciona el objeto a otro especificado a una velocidad "normal".
Controladores		
	ubicación	Sitúa al objeto en la ubicación especificada en el script correspondiente.

Automóvil		
Propiedades		
	frenar	Se activa cuando es preestablecido frenar o disminuir la velocidad. Ejemplo: cerca de un objeto tipo vehículo.
	vehículo	Propiedad que sirve para conocer el tipo del objeto. Ejemplo autos tipo "vehículo".
	choque	Se activa cuando se considera que hay choque. Ejemplo: colisión con un objeto tipo "vehículo".
	parar	Se activa cuando es preestablecido detenerse. Ejemplo: muy cerca de un objeto tipo "vehículo".
Sensores		
	pararRojo	Detecta la presencia de un objeto con material "rojo", frente a él.
	stopVehiculo	Detecta la presencia de un objeto con propiedad "vehículo", frente a él.
	pararPersona	Detecta la presencia de un objeto con propiedad "persona", frente a él.
	frenoProximidadPersona	Detecta la presencia de un objeto con propiedad "persona", frente a él.
	frenoProximidadVehiculo	Detecta la presencia de un objeto con propiedad "vehículo", frente a él.
	sensorSemaforoRojo	Detecta la presencia de un objeto con material "rojo", frente a él.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

	choqueBarreraVerde	Detecta la colisión con un objeto de material "rojo".
	choqueVehiculo	Detecta la colisión con un objeto de propiedad "vehículo".
	choquePersona	Detecta la colisión con un objeto de propiedad "persona".
	Always	Se activa al iniciar el programa.
	reiniciarUbicacion	Se activa al llegar al objetivo, la idea es volver a una ubicación deseada.
Actuadores		
	setStopTrue	Establece la propiedad "parar" en "verdadero".
	setStopFalse	Establece la propiedad "parar" en "falso".
	setFrenar	establece la propiedad "frenar" en "verdadero"
	setFrenarFalse	Establece la propiedad "parar" en "falso".
	setChoque	establece la propiedad "choque" en "verdadero"
	objetivoLento	Direcciona el objeto a otro especificado a una velocidad menor a la normal.
	objetivoNormal	Direcciona el objeto a otro especificado a una velocidad "normal".
Controladores		
	ubicación	Sitúa al objeto en la ubicación especificada en el script correspondiente.

Humanos:

Mujer		
Propiedades		
	choque	Se activa cuando se considera que hay choque. Ejemplo: colisión con un objeto tipo "vehículo".
	persona	Propiedad que sirve para conocer el tipo del objeto. Ejemplo Mujer tipo "persona".
Sensores		
	Always	Se activa al iniciar el programa.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

frenoSemaforo	Detecta la presencia de un objeto con material "Dorado" o "Azul" según corresponda, frente a él.
frenoVehiculo	Detecta la presencia de un objeto con propiedad "vehículo", frente a él.
choqueVehiculo	Detecta la colisión con un objeto de propiedad "vehículo".

Actuadores

setChoque	establece la propiedad "choque" en "verdadero"
avanzar	Direcciona el objeto a otro especificado a una velocidad.
accion_avanzar	Acciona el movimiento animado de caminar al objeto.
detener	detiene el objeto cuando se cumple una condición determinada

Controladores

ubicación	Sitúa al objeto en la ubicación especificada en el script correspondiente.

Planos:

Calzada	
Descripción	Objeto por donde circulan los vehículos

Vereda	
Descripción	Objeto por donde circulan los peatones.

Semáforo	
Descripción	Objeto representativo que señaliza el tránsito de vehículos y peatones. Cumple una función visual.
	Colores Verde, Amarillo y Rojo
Descripción	Los colores del semáforo se representan como objetos particulares mostrándolos y ocultándolos cuando corresponden, de acuerdo a una sincronización mediante el contador.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Propiedades

Contador	Variable que sirve para ser utilizada con segundero de un reloj.

Sensores

activarON	sensor que entrega un valor positivo, cuando el contador es igual al valor preestablecido
activarOFF	sensor que entrega un valor positivo, cuando el contador es igual al valor preestablecido
aumentarSegundo	Sensor que entrega pulsos positivos, constantemente con la finalidad de aumentar el contador en una unidad.
resetearContador	sensor que entrega un valor positivo, cuando el contador es igual al valor preestablecido

Actuadores

visibilityON	Actuador que permite al objeto ser visible.
visibilityOFF	Actuador que permite al objeto ser invisible.
add1	Actuador que aumenta el valor del contador en una unidad.
resetearContador	Restablece el contador en 0.

Barrera Semáforo

Descripción	Objeto que señaliza el tránsito de vehículos y peatones al aparecer en una ubicación predefinida.

Propiedades

Contador	Variable que sirve para ser utilizada con segundero de un reloj.

Sensores

activarON	sensor que entrega un valor positivo, cuando el contador es igual al valor preestablecido
activarOFF	sensor que entrega un valor positivo, cuando el contador es igual al valor preestablecido
aumentarSegundo	Sensor que entrega pulsos positivos, constantemente con la finalidad de aumentar el contador en una unidad.
resetearContador	sensor que entrega un valor positivo, cuando el contador es igual al valor preestablecido

Actuadores

add1	Actuador que aumenta el valor del contador en una unidad.
resetearContador	Restablece el contador en 0.

Puerta

Descripción	Funcionan como objetivo para objetos direccionados.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Propiedades		
	puerta	Usada para ser detectada por objetos que poseen actuadores de tipo colisión, como son los microbuses.

Malla de navegación.(Navmesh)	
Descripción	Objeto que sirve como ruta para vehículos o personas

Espejos retrovisores Microbús (Mirrors)	
Descripción	Cumple el objetivo, que el nombre describe. Revisar documentación en su página web[13]

Utilitarios:

Barrera	
Descripción	Objeto accesorio usado principalmente de ambientación para el escenario virtual.

Cámara	
Descripción	Sirve visualizar el escenario virtual desde una perspectiva específica.

Luces	
Descripción	Permite iluminar el escenario virtual

Focos	
Descripción	Objeto accesorio usado principalmente de ambientación para el escenario virtual.

[13] http://www.tutorialsforblender3d.com/Game_Engine/VideoTexture/VideoTexture_Mirror_1.html

Universidad del Bío-Bío. Red de Bibliotecas - Chile

8.2. Scripts Utilizados.

Ubicacion.py

El script Ubicacion.py, es utilizado para poder reubicar los actores que colisionan con objetos puerta, los que son usados como objetivos o meta evitando crear o destruir actores

Al ser llamada por un objeto reubica la posición del objeto mediante la propiedad localPosition de Blender traspasando los valores de los ejes como parámetros.

```
#def NombreNuevaUbicación(cont)
#    own =cont.owner
#    own.localPosition = (posición eje X, posición eje Y, posición eje Z)
```

El script posee distintas ubicaciones donde descritas, el propósito es que al aparecer los objetos no colisionen entre sí.

Semaforo.py

El script Semaforo.py, permite la aparición de los objetos barrera semáforos en la calzada con el propósito de ser detectado por los vehículos o peatones dependiendo del caso, por ello sólo se usa el eje Z.

Su funcionamiento es muy similar al Ubicacion.py, dado que al ser llamado por un objeto reubica la posición del objeto mediante la propiedad localPosition de Blender traspasando sólo el valor del eje Z como parámetro.

```
def NombreNuevaUbicacion(cont)
   own =cont.owner
   own.localPosition[2] = posición eje Z
```

Mirror.py

Este script es utilizado para poder simular el espejo retrovisor del microbús, está basado en el tutorial expuesto en el sitio tutorialsforblender3d[14]. El que describe la forma con la cual implementar un objeto que tenga la capacidad de visualizar en su textura una escena.

[14] http://www.tutorialsforblender3d.com/Game_Engine/VideoTexture/VideoTexture_Mirror_1.html

Universidad del Bío-Bío. Red de Bibliotecas - Chile

9. CONCLUSIONES

Posterior a desarrollo del proyecto realizado, y presentado en este documento se concluye que el desarrollo de éste es beneficioso para el LEU, debido a que aporta y respalda la metodología implementada, además permite el desarrollo de escenarios y objetos que simulen el tránsito de vehículos y peatones como se requería desde el principio.

Adicionalmente se concluye que el desarrollo de escenarios se debe ajustar a la capacidades del software a utilizar, en este caso, Blender permite simular el comportamiento de objetos y posee distintas funciones que ayudan al cumplimiento de dichos comportamientos, pero al intentar simular el comportamiento humano ya sea al interactuar como peatón o conductor, existen muchas variables que se evalúan mediante condiciones, las cuales resultan triviales para cada uno, pero al intentar simularlas se deben dejar descritas, y a medida que se agregan, complejizan la lógica para los objetos y el entendimiento de ésta para los usuarios por ello es fundamental llevar un registro que respalde la lógica implementada.

El desarrollo de este proyecto planteó desafíos fuera de lo común, lo cual resultó provechoso, dado que se trato de un proyecto de investigación y desarrollo, solucionando los problemas ajustándose a las funcionalidades propias de Blender, empleando los conocimientos y habilidades adquiridas durante la estancia en la carrera de Ingeniería (E) en Computación e Informática en la Universidad del Bío-Bío, intentando implementar dichas soluciones de una manera ingeniosa y más simple posible, siempre pensando en la amplia gama usuarios.

Universidad del Bío-Bío. Red de Bibliotecas - Chile

10. REFERENCIA BIBLIOGRÁFICA

Introducción práctica a la realidad virtual,

Juan Carlos Parra Márquez, 1991

Alfa Gaviota UBB wiki

http://leu.servicios.ubiobio.cl/alfa_wiki/

Blender API documentation

http://www.blender.org/documentation/blender_python_api_2_68_release/contents.html

Blender - Game Engine

http://wiki.blender.org/index.php/Dev:ES/Ref/Release_Notes/2.66/Game_Engine#Mandos_de_juego_con_Python

Blender - Real Time Mirror

http://www.tutorialsforblender3d.com/Game_Engine/VideoTexture/VideoTexture_Mirror_4.html

SISTEMA DE SIMULACIÓN EN 3D PARA EL ROBOT SCORBOT ER-VPLUS

Karina Pilar Cid Cifuentes, 2007

Universidad del Bío-Bío. Red de Bibliotecas - Chile

11. ANEXO

Script utilizados

Semaforo.py

```
import GameLogic as GL
import Rasterizer as R

#Para crear una nueva ubicación debe seguir la secuencia
#def NombreNuevaUbicacion(cont)
#    own =cont.owner
#    own.localPosition[2] = posición eje Z


def ON(cont):
  own = cont.owner
  own.localPosition[2]= 0

def OFF(cont):
  own = cont.owner
  own.localPosition[2]= -4
```

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Ubicacion.py

```
import GameLogic as GL
import Rasterizer as R

####COORDENADAS PARA LA UBICACIÓN DE OBJETOS##
#
#Para crear una nueva ubicación debe seguir la secuencia
#def NombreNuevaUbicación(cont)
#    own =cont.owner
#    own.localPosition = (posición eje X, posición eje Y, posición eje Z)
#


##VEHICULOS##

##Pista Derecha##
def Collao(cont):
  own = cont.owner
  own.localPosition = (223,-74,1.6)

##Pista Izquierda##
def Collao2(cont):
  own = cont.owner
  own.localPosition = (221,-80,1.6)

##Pista Derecha##
def CollaoChacabuco(cont):
  own = cont.owner
  own.localPosition = (127,-96,1.6)

##Pista Izquierda##
def CollaoChacabuco2(cont):
  own = cont.owner
  own.localPosition = (122,-93,1.6)

def Carrera(cont):
  own = cont.owner
  own.localPosition = (-133,-44,1.6)
```

Universidad del Bío-Bío. Red de Bibliotecas - Chile

```
def Irarrazaval(cont):
  own = cont.owner
  own.localPosition = (-63,-73,1.6)


def IrarrazavalBaquedano(cont):
  own = cont.owner
  own.localPosition = (70,156,1.6)



  ##PEATONES##
def PeatonIrarrazaval1(cont):
  own = cont.owner
  own.localPosition = (16,28,1.4)


def PeatonIrarrazaval2(cont):
  own = cont.owner
  own.localPosition = (-4,-4,1.4)

#Altura paradero hospital
def PeatonIrarrazaval3(cont):
  own = cont.owner
  own.localPosition = (2,81,1.4)

def PeatonCarrera1(cont):
  own = cont.owner
  own.localPosition = (-66,-16,1.4)
```

Universidad del Bío-Bío. Red de Bibliotecas - Chile

Mirror.py

```
##########################################################
#
#   Mirror.py       Blender 2.5
#
#   Tutorial for using Mirror.py can be found at
#
#   www.tutorialsforblender3d.com
#
#   Released under the Creative Commons Attribution 3.0 Unported License.
#
#   If you use this code, please include this information header.
#
##########################################################

# import bge module
import bge

# get the current controller
controller = bge.logic.getCurrentController()

# get object script is attached to
obj = controller.owner

# check to see variable Mirror has been created
if "Mirror" in obj:

        # update the mirror
        obj["Mirror"].refresh(True)

# if variable Mirror hasn't been created
else:

        # get current scene
        scene = bge.logic.getCurrentScene()

        # get the mirror material ID
        matID = bge.texture.materialID(obj, "MA" + obj['material'])
################################################
```

Universidad del Bío-Bío. Red de Bibliotecas - Chile

```
        # get object list
        objList = scene.objects

        # get camera named Camera_1
        cam_1 = objList["Camera2"]

        # make cam_1 the active camera
        #scene.active_camera  = cam_1

        # get the active camera
        #cam = scene.active_camera
        cam = cam_1

#####################################################

        # texture channel
        if 'channel' in obj:

                # set texture channel
                texChannel = obj['channel']

        else:

                # use texture channel 1
                texChannel = 0

        # get the mirrortexture
        mirror = bge.texture.Texture(obj, matID, texChannel)

        # get the mirror source
        mirror.source = bge.texture.ImageMirror(scene, cam, obj, matID)

        # save mirror as an object variable
        obj["Mirror"] = mirror
```