22AIE442 Labsheet 5

Name: Aniketh Vijesh
Roll No: AM.EN.U4AIE22009

1. World Plugin

WorldPluginExample.cpp

```
#include <gazebo/gazebo.hh>
namespace gazebo {
class WorldPluginExample : public WorldPlugin {
public:
    WorldPluginExample() : WorldPlugin() {
        printf("Hello, World Plugin!\n");
    }
    void Load(physics::WorldPtr _world,
sdf::ElementPtr _sdf) override {}
};
GZ_REGISTER_WORLD_PLUGIN(WorldPluginExample)
}
```

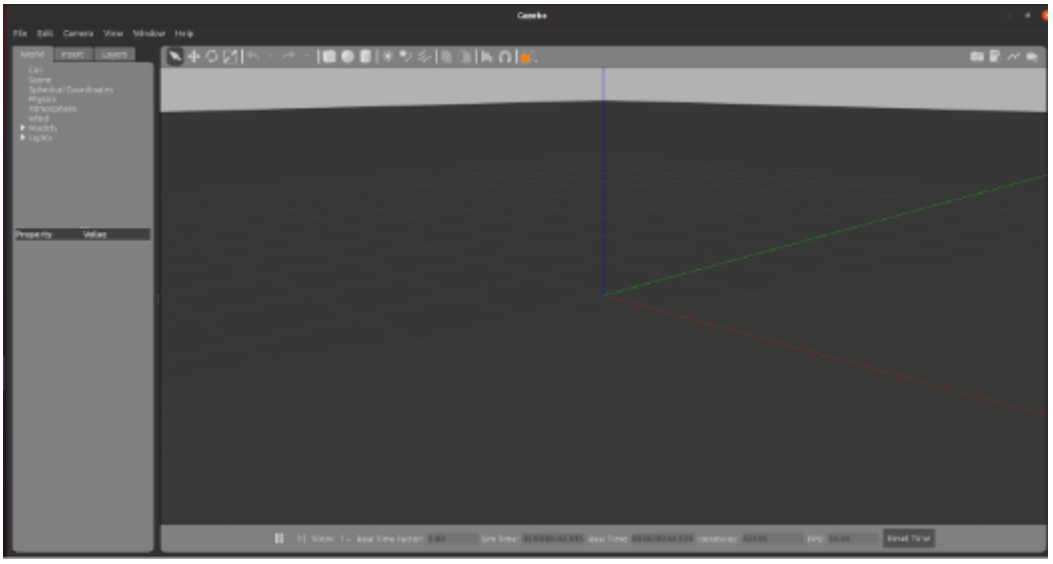
CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0 FATAL_ERROR)
find_package(gazebo REQUIRED)
include_directories(${GAZEBO_INCLUDE_DIRS})
add_library(WorldPluginExample SHARED
WorldPluginExample.cpp)
target_link_libraries(WorldPluginExample
${GAZEBO_LIBRARIES})
```

world_plugin_test.sdf

```
<?xml version="1.0" ?>
```

```
<sdf version="1.6">
  <world name="default">
    <plugin name="world_plugin"
filename="libWorldPluginExample.so"/>
  </world>
</sdf>
```



2. Model Plugin

ModelPluginExample.cpp

```
#include <gazebo/gazebo.hh>
#include <gazebo/physics/physics.hh>
namespace gazebo {
class ModelPluginExample : public ModelPlugin {
public:
    ModelPluginExample() {}
    void Load(physics::ModelPtr _model,
sdf::ElementPtr _sdf) override {
        printf("Model Plugin Loaded: %s\n",
_model->GetName().c_str());
    }
```

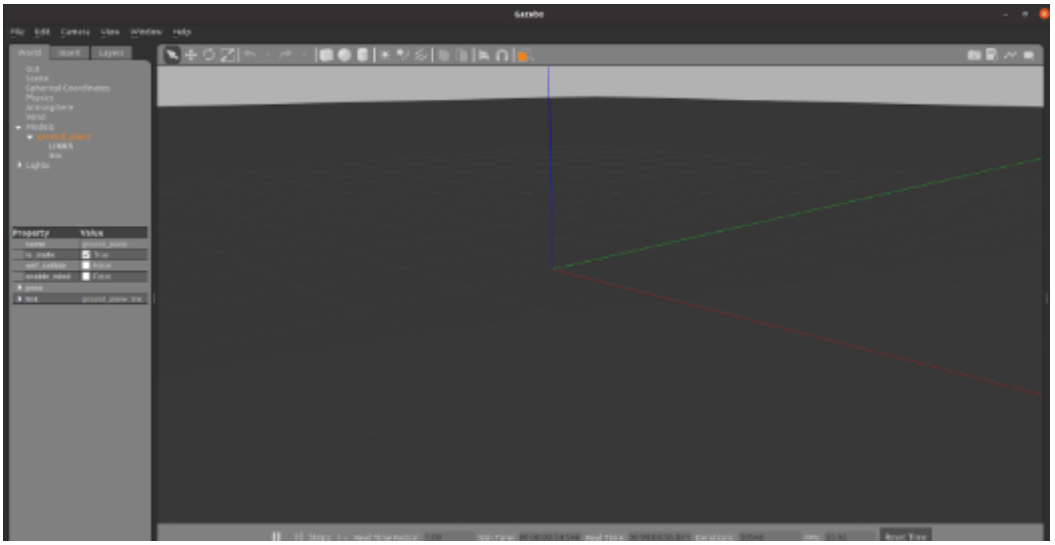
```
};
GZ_REGISTER_MODEL_PLUGIN(ModelPluginExample)
}
```

CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0 FATAL_ERROR)
find_package(gazebo REQUIRED)
include_directories(${GAZEBO_INCLUDE_DIRS})
add_library(ModelPluginExample SHARED
ModelPluginExample.cpp)
target_link_libraries(ModelPluginExample
${GAZEBO_LIBRARIES})
```

model_plugin_test.sdf

```
<?xml version="1.6" ?>
<sdf version="1.6">
  <world name="default">
    <model name="test_model">
      <link name="link">
        <collision name="collision">
          <geometry><box><size>1 1
1</size></box></geometry>
        </collision>
        <visual name="visual">
          <geometry><box><size>1 1
1</size></box></geometry>
        </visual>
      </link>
      <plugin name="model_plugin"
filename="libModelPluginExample.so"/>
    </model>
  </world>
</sdf>
```



3. Sensor Plugin

SensorPluginExample.cpp

```
#include <gazebo/gazebo.hh>
#include <gazebo/sensors/sensors.hh>
namespace gazebo {
class SensorPluginExample : public SensorPlugin {
public:
    SensorPluginExample() {}
    void Load(sensors::SensorPtr _sensor,
sdf::ElementPtr _sdf) override {
        printf("Sensor Plugin Loaded: %s\n",
_sensor->Name().c_str());
    }
};
GZ_REGISTER_SENSOR_PLUGIN(SensorPluginExample)
}
```

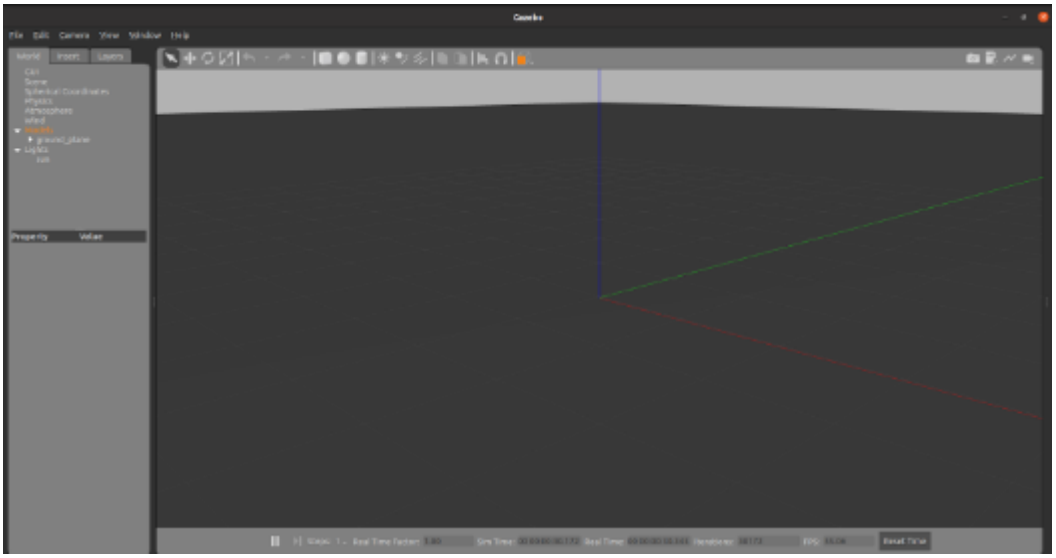
CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0 FATAL_ERROR)
find_package(gazebo REQUIRED)
```

```
include_directories(${GAZEBO_INCLUDE_DIRS})
add_library(SensorPluginExample SHARED
SensorPluginExample.cpp)
target_link_libraries(SensorPluginExample
${GAZEBO_LIBRARIES})
```

sensor_plugin_test.sdf

```
<?xml version="1.0" ?>
<sdf version="1.6">
  <world name="default">
    <sensor name="test_sensor" type="camera">
      <plugin name="sensor_plugin"
filename="libSensorPluginExample.so"/>
    </sensor>
  </world>
</sdf>
```



4. System Plugin

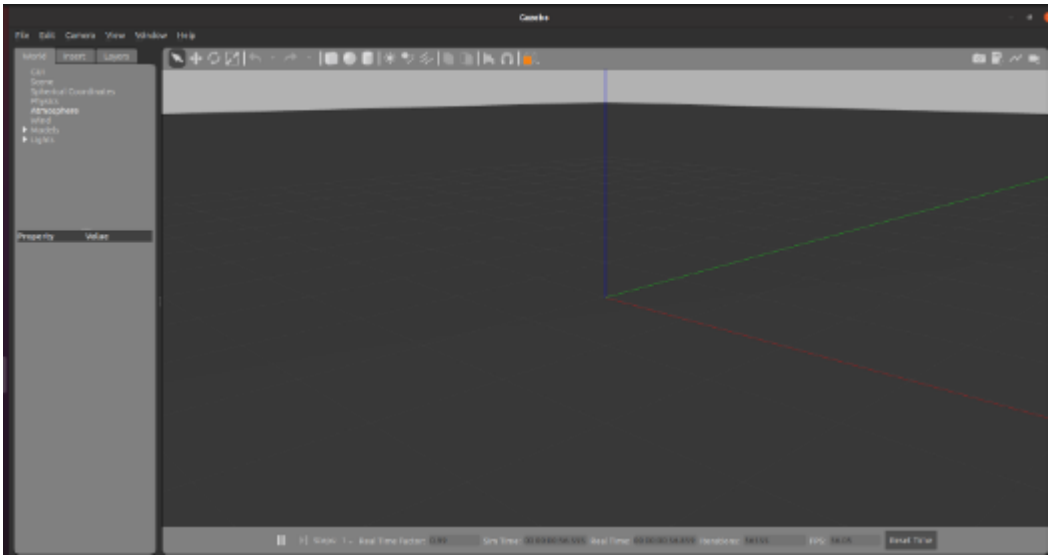
SystemPluginExample.cpp

```
#include <gazebo/gazebo.hh>
namespace gazebo {
class SystemPluginExample : public SystemPlugin {
```

```
public:
    SystemPluginExample() {}
    void Load(int argc, char **argv) override {
        printf("System Plugin Loaded\n");
    }
};
GZ_REGISTER_SYSTEM_PLUGIN(SystemPluginExample)
}
```

CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0 FATAL_ERROR)
find_package(gazebo REQUIRED)
include_directories(${GAZEBO_INCLUDE_DIRS})
add_library(SystemPluginExample SHARED
SystemPluginExample.cpp)
target_link_libraries(SystemPluginExample
${GAZEBO_LIBRARIES})
```



5. Visual Plugin

VisualPluginExample.cpp

```
#include <gazebo/gazebo.hh>
#include <gazebo/rendering/rendering.hh>
namespace gazebo {
```

```
class VisualPluginExample : public VisualPlugin {
public:
    VisualPluginExample() {}
    void Load(rendering::VisualPtr _visual,
sdf::ElementPtr _sdf) override {
        printf("Visual Plugin Loaded: %s\n",
_visual->Name().c_str());
    }
};
GZ_REGISTER_VISUAL_PLUGIN(VisualPluginExample)
}
```

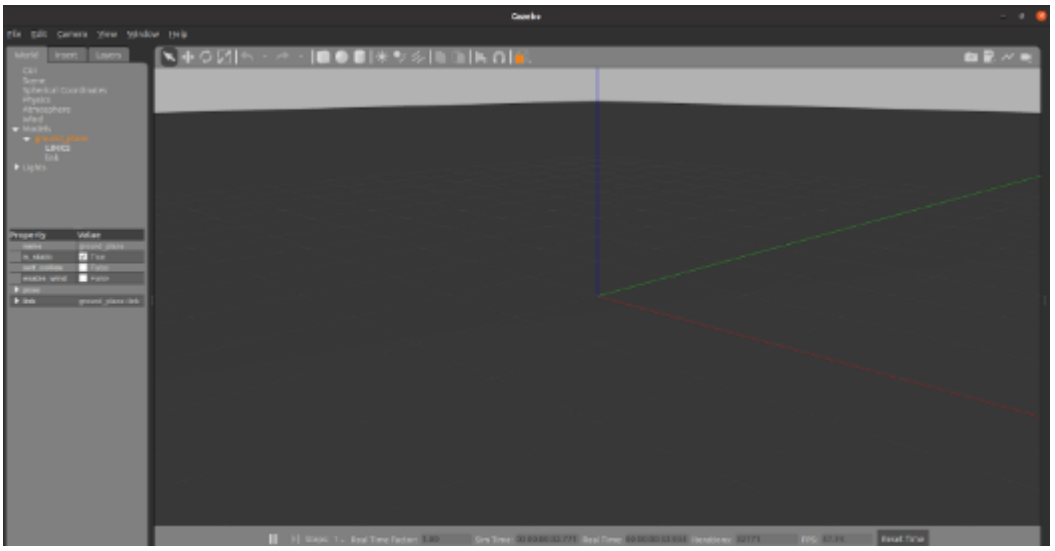
CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0 FATAL_ERROR)
find_package(gazebo REQUIRED)
include_directories(${GAZEBO_INCLUDE_DIRS})
add_library(VisualPluginExample SHARED
VisualPluginExample.cpp)
target_link_libraries(VisualPluginExample
${GAZEBO_LIBRARIES})
```

visual_plugin_test.sdf

```
<?xml version="1.0" ?>
<sdf version="1.6">
  <world name="default">
    <model name="visual_model">
      <link name="link">
        <visual name="test_visual">
          <geometry><box><size>1 1
1</size></box></geometry>
          <plugin name="visual_plugin"
filename="libVisualPluginExample.so"/>
        </visual>
      </link>
```

```
    </model>
  </world>
</sdf>
```



6. GUI Plugin

GUIExampleSpawnWidget.cc

```
#include <sstream>
#include <gazebo/msgs/msgs.hh>
#include "GUIExampleSpawnWidget.hh"

using namespace gazebo;

GZ_REGISTER_GUI_PLUGIN(GUIExampleSpawnWidget)

GUIExampleSpawnWidget::GUIExampleSpawnWidget() :
GUIPlugin() {
  this->counter = 0;
  this->setStyleSheet("QFrame { background-color :
rgba(100, 100, 100, 255); color : white; }");
```

```
  QHBoxLayout *mainLayout = new QHBoxLayout;
  QFrame *mainFrame = new QFrame();
  QVBoxLayout *frameLayout = new QVBoxLayout();
  QPushButton *button = new QPushButton(tr("Spawn
Sphere"));
  connect(button, SIGNAL(clicked()), this,
SLOT(OnButton()));
  frameLayout->addWidget(button);
  mainFrame->setLayout(frameLayout);
  mainLayout->addWidget(mainFrame);
  frameLayout->setContentsMargins(0, 0, 0, 0);
  mainLayout->setContentsMargins(0, 0, 0, 0);
  this->setLayout(mainLayout);
  this->move(10, 10);
  this->resize(120, 40);
  this->node = transport::NodePtr(new
transport::Node());
  this->node->Init();
  this->factoryPub =
this->node->Advertise<msgs::Factory>("~/factory");
}

GUIExampleSpawnWidget::~GUIExampleSpawnWidget() {}

void GUIExampleSpawnWidget::OnButton() {
  msgs::Model model;
  model.set_name("plugin_unit_sphere_" +
std::to_string(this->counter++));
  msgs::Set(model.mutable_pose(),
ignition::math::Pose3d(0, 0, 1.5, 0, 0, 0));
  msgs::AddSphereLink(model, 1.0, 0.5);
  msgs::Factory msg;
  msg.set_sdf(model.SerializeAsString());
  this->factoryPub->Publish(msg);
}
```

GUIExampleSpawnWidget.hh

```
#ifndef _GUI_EXAMPLE_SPAWN_WIDGET_HH_
#define _GUI_EXAMPLE_SPAWN_WIDGET_HH_

#include <gazebo/common/Plugin.hh>
#include <gazebo/gui/GuiPlugin.hh>
#include <gazebo/transport/transport.hh>

namespace gazebo {
    class GAZEBO_VISIBLE GUIExampleSpawnWidget :
public GUIPlugin {
      Q_OBJECT

      public: GUIExampleSpawnWidget();
      public: virtual ~GUIExampleSpawnWidget();

      protected slots: void OnButton();

      private: unsigned int counter;
      private: transport::NodePtr node;
      private: transport::PublisherPtr factoryPub;
    };
}
#endif
```

CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0 FATAL_ERROR)
find_package(gazebo REQUIRED)
find_package(Qt5Widgets REQUIRED)

include_directories(${GAZEBO_INCLUDE_DIRS}
${Qt5Widgets_INCLUDE_DIRS})
```

```
link_directories(${GAZEBO_LIBRARY_DIRS})

add_library(gui_example_spawn_widget SHARED
GUIExampleSpawnWidget.cc)
target_link_libraries(gui_example_spawn_widget
${GAZEBO_LIBRARIES} ${Qt5Widgets_LIBRARIES})
```