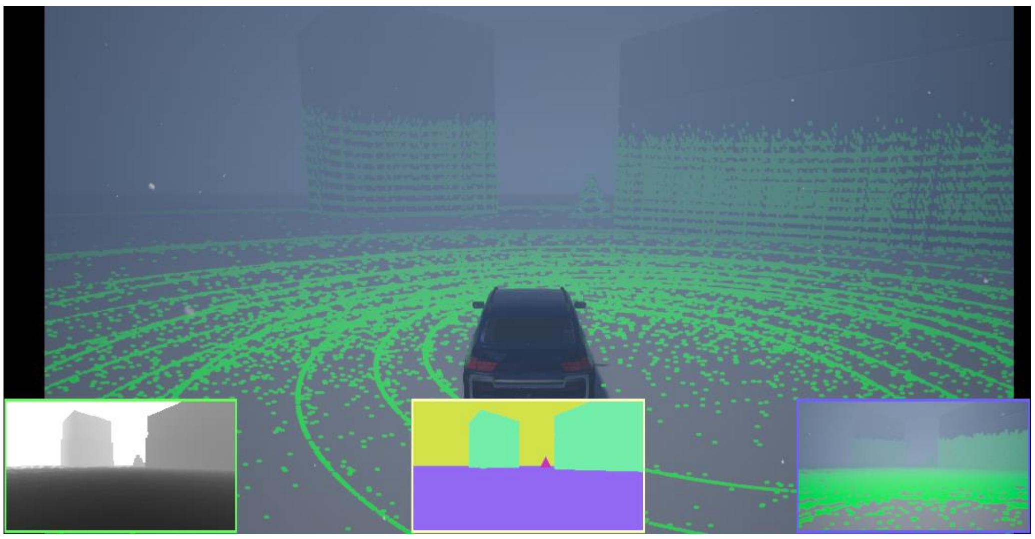
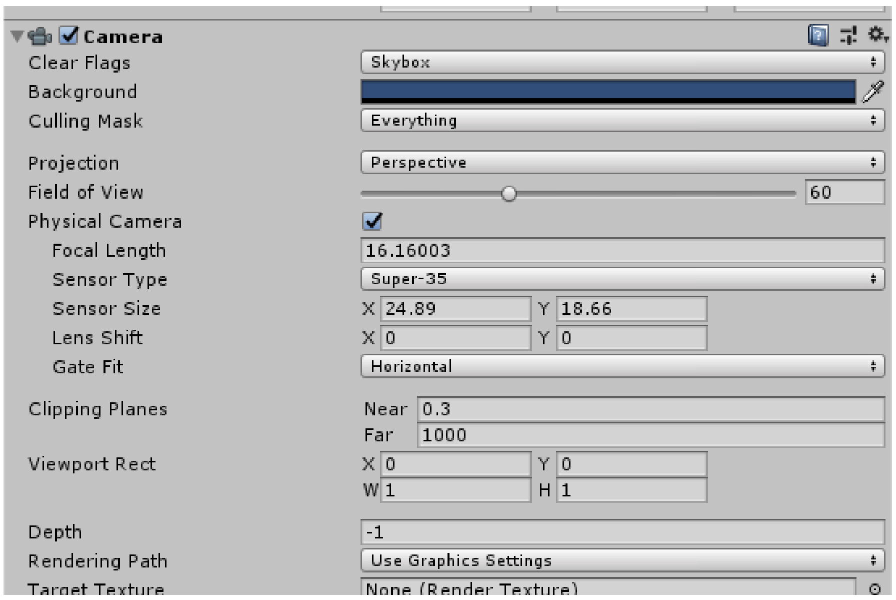
Autonomous vehicles Making Decisions

Introduction: The AMD LIDAR Calibration system was designed to combat the uncertainty created in autonomous vehicle LIDAR sensors in extreme weather conditions. The system aims to correct erroneous point clouds produced when objects and particles impede the light rays traveling from the LIDAR sensor. To prevent hazardous decision-making based on the presence of this interference, the system provides assurance that erroneous data is to be ignored. The system tracks the distance between itself and the nearest vehicle, and prevents the unnecessary emergency braking that would occur if the system were to suddenly detect an object in its path.

Simulation: The testing environment for this project is currently built using the Unity game engine. Non-target cars and weather inside the engine are controlled using a predetermined pattern, while the target car is controlled using an exposed Python API. This will allow for future iterations of the project to calibrate the sensors and then control the target car in real time to test the calibration effectiveness. Outside of a simulated physical camera, the simulation also outputs LIDAR point cloud data, depth views, and object segmentation views.



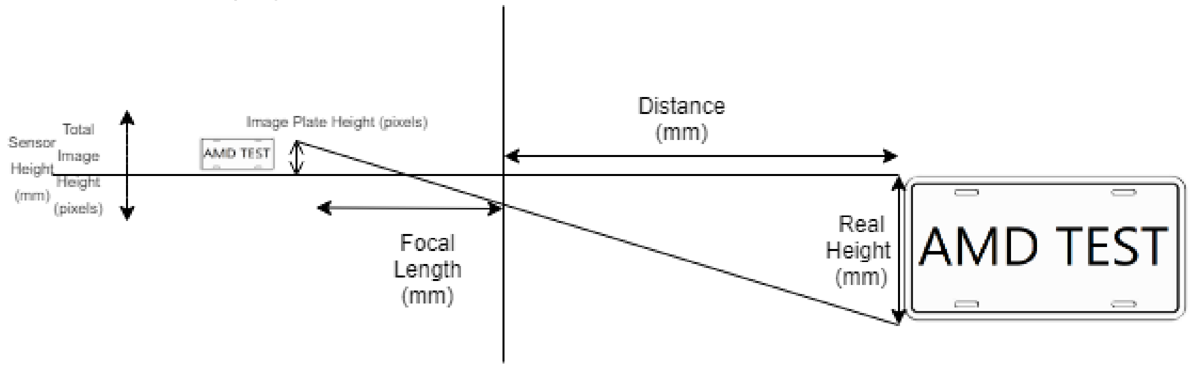
Camera Calibration: The internal camera parameters were determined by utilizing the physical camera feature of the simulation engine. This enabled us to retrieve the sensor size and focal length, which are pivotal values in our distance estimation.



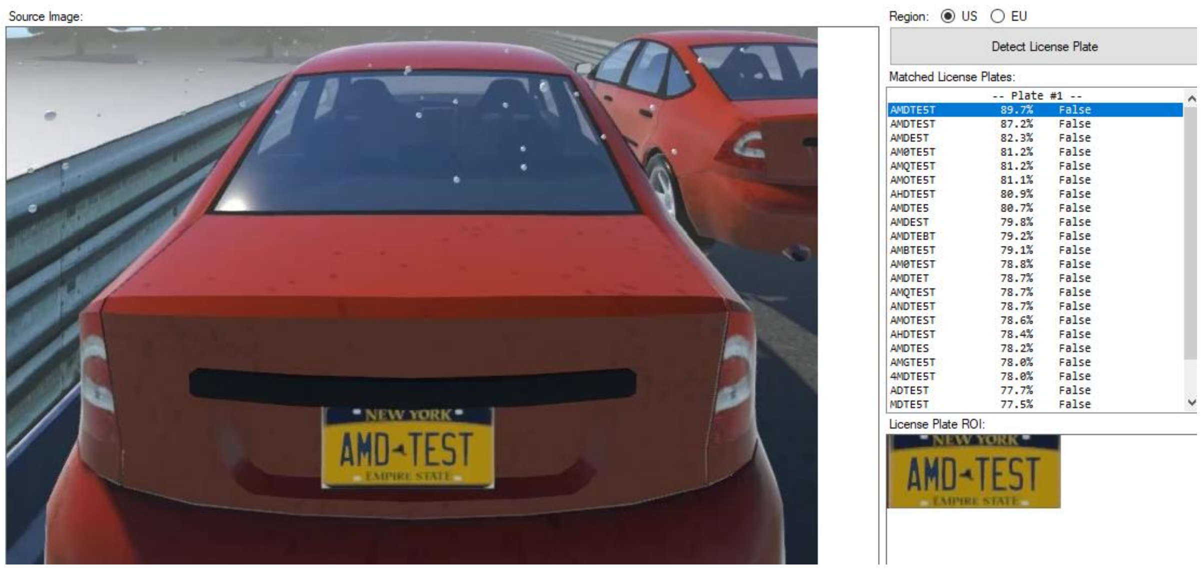
Camera

Clear Flags	Skybox
Background	
Culling Mask	Everything
Projection	Perspective
Field of View	60
Physical Camera	✓
Focal Length	16.16003
Sensor Type	Super-35
Sensor Size	X 24.89 Y 18.66
Lens Shift	X 0 Y 0
Gate Fit	Horizontal
Clipping Planes	Near 0.3 Far 1000
Viewport Rect	X 0 Y 0 W 1 H 1
Depth	-1
Rendering Path	Use Graphics Settings
Target Texture	None (Render Texture)

Distance Estimation: Using the internal parameters of the camera, as well as a pinhole projection model and the principle of triangle similarity, we infer the distance of the object from the centre of projection of the lens.



OpenALPR: This open-sourced library was a foundational technology of this project. Using OpenCV and Tesseract, this library detects license plates in images. Using a test set which the model has been trained to use, this flexible library can be adapted to any set of plates and will function as expected.



```
            -- Plate #1 --
AMDTEST         89.7%     False
AMDTEST         87.2%     False
AMDEST          82.3%     False
AM0TEST         81.2%     False
AMQTEST         81.2%     False
AMOTEST         81.1%     False
AHDTEST         80.9%     False
AMDTES          80.7%     False
AMDEST          79.8%     False
AMDTEBT         79.2%     False
AMBTEST         79.1%     False
AM0TEST         78.8%     False
AMDTET          78.7%     False
AMQTEST         78.7%     False
ANDTEST         78.7%     False
AMOTEST         78.6%     False
AHDTEST         78.4%     False
AMDTES          78.2%     False
AMGTEST         78.0%     False
4MDTEST         78.0%     False
ADTEST          77.7%     False
MDTEST          77.5%     False
```

Performance Optimization: Running time is a huge concern in this project, as mere seconds can draw the line between system success and failure. To help reduce the running time of this algorithm, a cropped image detection thread is run in parallel to a thread analyzing the full view of the camera. The cropped view yielded a 90% running time reduction, greatly increasing the speed of the program.