# Demining the 3D world

## 1 Introduction

This project aim at programming a truck equipped with a metal detector on a lateral arm to autonomously detect mines in a specific environment.

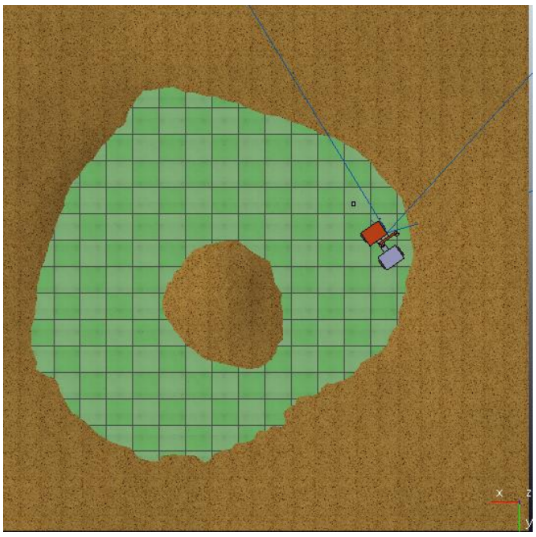

Figure 1: Simulation environment

The mines are known to be at the external interface with the plane ground in green and the mud mounts in brown. The truck has the following sensors which position can be modified:

Table 1: Available sensors

| Sensor | Output |
|---|---|
| Camera | Color image |
| Kinect | 3-D point cloud |
| Laser | 2-D point cloud |
| Metal Detector | Real number |

Here are the objective the system must complete:

- Maintain the metal detector the nearest possible to the interface
- Maintain the metal detector above the interface so that it never touches it
- Maintain the truck near the interface and make it autonomously drive

We split the three objectives into two controls:

- The first control maintain the metal detector above the interface. To do so, there is one discriminative information that answers both the question of where is the interface and what is its height: the ground height. Since the truck already moves along the interface, it is feasible to restrict the arm movement only to lateral movement. Then the only height discrimination we need is a 2D discrimination along the arm axis. To collect these information, the 2D laser appears to be a relevant sensor and it is placed a little before the arm axis.

- The second control makes the truck go forward while maintaining it at a fixed distance from the interface so that the metal detector can always reach it. To do so, we use a linear control on the truck position with respect to the interface. We also want the truck to be able to anticipate its movements given that it may encounter steep turns and that its shift capacity is mechanically limited. So we need to detect the interface limit in front of the truck. To do so, we use againheight discrimination and we collect the data points through the Kinect. A relevant position for the Kinect is at the truck front looking a bit to the left since the interface is at its left.

Finally, the final position of the laser and the Kinect are as follow:

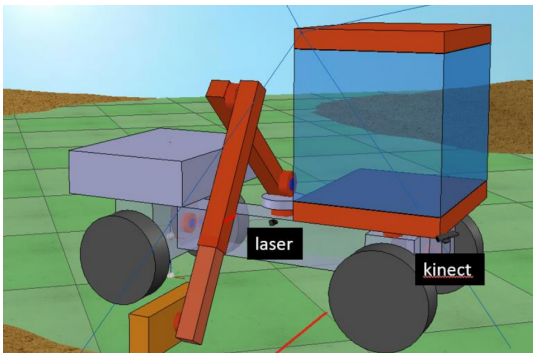


Figure 2: Sensor position

## 2 Arm control

Given the regularity of the simulation world, we use a threshold to detect the point from which the interface starts in the arm direction. We set the threshold by observing the height evolution along the arm axis and choosing a height in the increasing window of the line. After several tries, we find that a value of for the threshold allows the arm to above the interface: $threshold = 1cm$. However, once the environment is different, it would require us to change the threshold so this model is not portable. Another problem we have to deal with is the unexpected presence of obstacles in the laser view. The figure 3 below show an example of such situation.

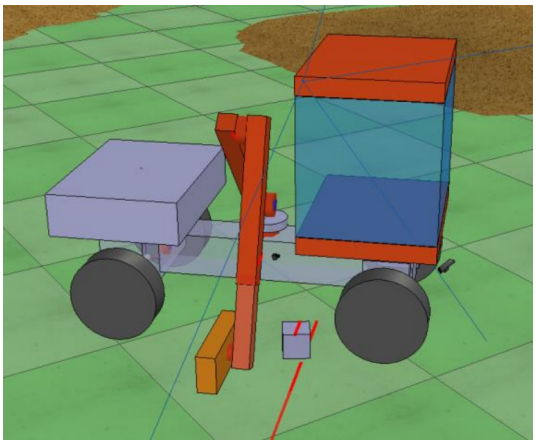

Figure 3: Obstacle disturbance

This would result in undesirable detection of a point above the threshold and then a bad arm position. We deal with this problem by taking advantage of the environment so once again this solution is not portable but works for this case. We observe that the interface is continuous and relatively regular, so the presence of an obstacle would result in an outlier detected point. So we add condition on the laser detection to ignore outlier point above the height threshold. The figure below 4 show which point we detect and the outlier data points found by the laser on the arm truck.

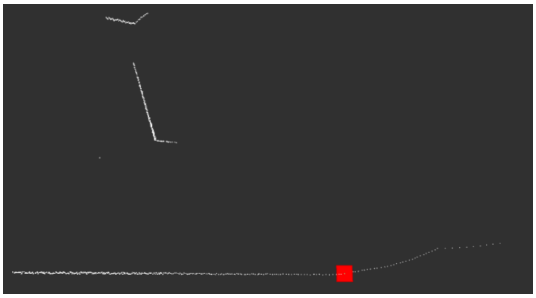

Figure 4: Obstacle disturbance

We also have to position the tool at a height such that it does not touch the interface. The laser give us the interface height and we position the tool at a height superior of $delta = 8mm$, delta being a parameter.

Once we have the final position of the tool, we change its position by position control. The arm as it automatically computes its movement from its original position to the target position via a provided python script. This results in unnatural movement of the arm as it does not go straight to its position. A speed control would be smoother but has not been implemented in this project.

As for the laser sampling rate, it both depends on the truck velocity and the metal detector sampling rate. The higher the velocity of the truck, the higher the laser sampling frequency must be to get a sampling as uniform as possible. But the laser sampling rate should not exceed the metal detector sampling rate to ensure that we do not miss any mining data. The first parameter to set is the truck velocity and then the laser and metal detector sampling rate depending on the regularity of the interface and the desired precision of the detection. We set the metal detector sampling rate at a higher frequency that the laser one to get new samples not only when the tool laterally moves from

the arm movements (induced by the laser sampling) but also to get new samples as the truck moves.

The bounds on the laser rate as defined by:

- Upper bound
  - Its mechanical limit
  - The metal detector mechanical limit because we want its sampling rate to be higher than the laser one
- Lower bound
  - The truck velocity
  - The environment regularity

We have not done experiment to find the values that allow the maximum coverage and have stayed with default sampling frequency $f = 20Hz$. We have then set the truck velocity to $v = 0.8m/s$ which gives a window of $40cm$ where there is no arm control and no metal detection. Given the regularity of the interface and how the truck follows the interface, we can afford such a window. As for the metal detector, given that the signal is proportional to the proximity of the detector the mine, we are sure to get signal when we are near the mine, even if we do not get signal exactly when we are above the mine. To compensate for this imprecision, we average the signal as explained in section 4.

# 3 Control truck

The truck position is controlled by its distance to the interface and its angular movement with respect to the interface direction. That is why we first need to get the points that form the interface. To do so, we first build an elevancy map:

- We discretize the world in cells and compute the average height of the cell
  - If the height is bigger than a threshold, the cell is the brown region an we colour it in grey
  - Else it is the green region and we colour it in white
- We use the the OpenCV library to extract points to constitute the contours between the different colours regions.
- We filter the contour to end only with the points between the white and the grey regions (the last image has a rotation of $\frac{\pi}{4}$.

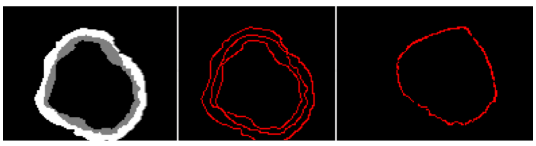

Figure 5: Contour detection

We want to keep the truck at a constant distance $y_{stat}$ from the interface. That is why we use this simplified version of control:

$$v_r = v$$
$$v_\theta = k_1 y + k_2 \theta$$

Where $k_1$ , $k_2$ are the control gain to set, $y = y_{stat} - y_{current}$ where $y_{current}$ is the current distance of the truck from the interface and we set $y_{stat} = 3.0$ , $\theta$ is the angle between the truck direction and the interface direction. The equilibrium is obtained for $\theta = 0$ and $y = 0$.

Since we want the truck to anticipate its movements, all the measurement will be based on a hypothetical future position of the truck if there was no control. The figure below describes these measurements.

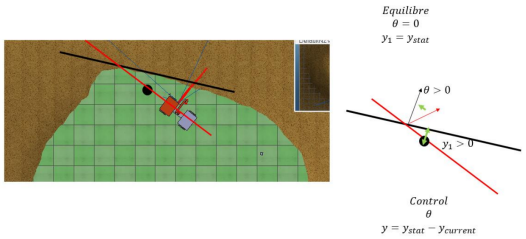


Figure 6: Measurements method

We define a look ahead point at a distance `look_ahead` from the front of the truck along its axis (black circle). We then compute the orthogonal projection of this point on the interface and compute the tangent of the interface at this projection (black line): this the control direction. We compute the distance fromthe look ahead point to this line and get y current . We also compute the angle between this line and the truck axis (red line) to get $\theta$.

Once we know the global distribution of $\theta$ and $y$, we set $k_1, k_2$ so that the control is saturated for the truck positions that most need control. Given the environment distribution, we define the position for which the angle control must be maximal by the red circle, and the position for which the distance control must be maximal by the blue circle. (FIgure 7)

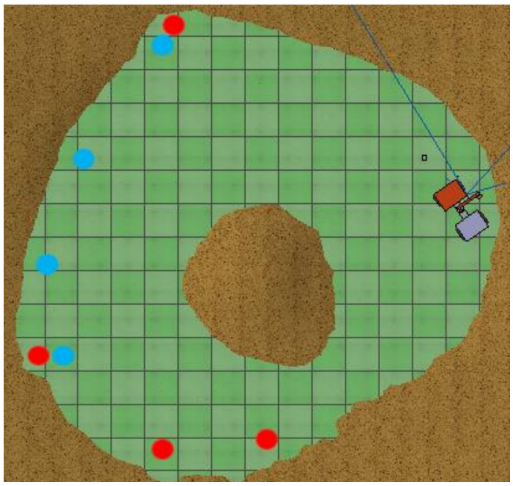

Figure 7: Measurements method

We found maximum gap values of $\theta = \frac{\pi}{2}$ and $y = 3.0$ at the red positions and blue positions. We see when he arrives at the left up corner so we first set $k_1 = 1$. We then compute an order of $k_2$ from the extremum found before. We represent the situation in the figure below. Given that we have set the truck velocity $v = 0.8m/s$ and that the time discretization for the truck is $dt = \frac{1}{20}s$

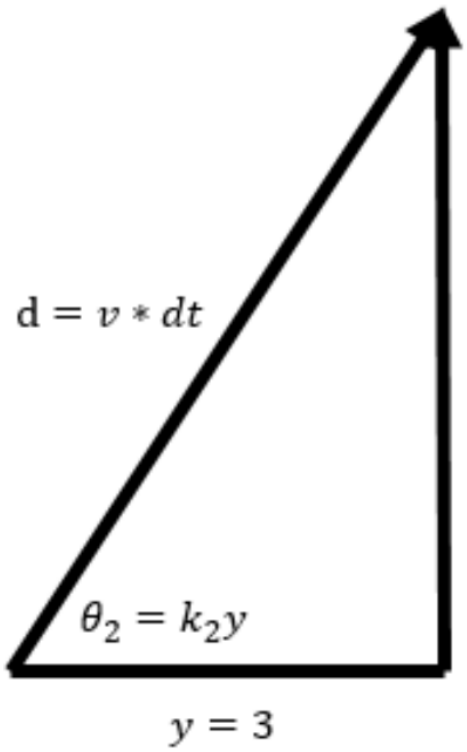


Figure 8: $\theta$ computation

We have $cos(\theta_2) = \frac{y}{d}$ so $k_2 = \frac{1}{y} acos \frac{d}{y} = 0.5$ with our setting. Once we plug in this value, we get the desired control such that the truck follows the interface variation.

Now that we have found relevant values for the gain, we improve the truck control by defining the degree of anticipation through the look ahead parameter. The key position where we most need to anticipate is the upper right corner and we find the relevant `look_ahead` value by trial and error: while the truck does not turn soon enough, we increase the `look_ahead`. Finally, we find that that `look_ahead = 2.5m` allows the truck to take decision soon enough to get through steep turns.

# 4 Mine detection

Given the metal detector sampling frequency, we must gather several detection samples to get an accurate position computation. To do so, we keep track of position for which the mine detector returns high value above `mine_threshold`. We store both the position and the returned value. Since the mines are apart, we get five clusters of points weighted by the trust we can put in these points. The nearer to 1, the more trustful it is so we compute the barycentre of the points in each cluster to get the mine position.

The `mine_threshold` parameter depends on the size of the window around the mine in which we want to start the detection. In our case, we have observed that when `mine_threshold = 0.9` allow to start the detection 50cm before and 50 cm after the mine.

# 5 Conclusion

The first conclusion we get from this project is the importance of data visualization especially geometric results. We had to deal with computation errors and the hardest part was first to find these errors. Once we have implemented visualization tools, we could rapidly correct these errors.

The second conclusion we can draw is the poor reliability of threshold: there prevent the model to be portable to other environment, they do not guarantee correct results since any bogus point can lead to unpredictable results. That is why it would be preferable to use another method to detect the interface via the laser. We could process the image we get from the camera to implement a "linear" separator between green points and brown points.

The third conclusion we get is the methodology of the gain parameter of the control loop. By studying only extreme situations, we could efficiently define order of values for the gain parameter and then lightly adjust them to get the relevant values.