Multi-Camera Fusion in a Smart Intersection

RUTGERS
WINLAB | Wireless Information Network Laboratory
Peter Wilmot, Jesse Lerner, Varun Kota, Yue Wang, Arnuv Batra

Introduction

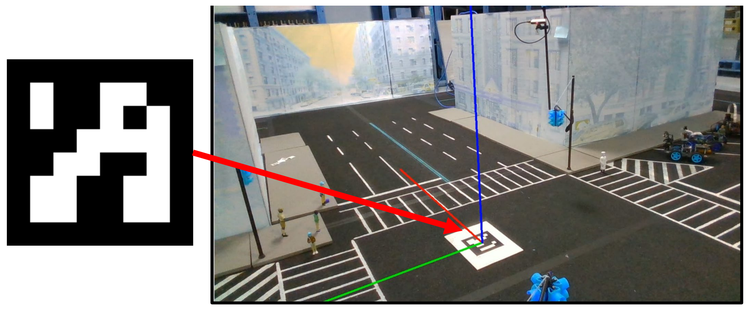
The way we represent 3D environments on computers is a format called a "Point Cloud," which is essentially a list of colored points with their positions in space. This format is used in robotics that need knowledge of their surroundings, such as self-driving cars. We record these Point Clouds using 3D Cameras, but these cameras can only record in one direction. A self driving car approaching the intersection needs to know what is happening at every angle, so these Point Clouds need to be unified for the car to know where it is in relation to other cars approaching the intersection.

ArUco Tags

What are ArUco Tags?

ArUco Tags look like simplified QR codes.

A camera can determine its position by reading a tag and, by extension, its position relative to other cameras.



The Idea

Camera 1 sees the ArUco tag and identifies its position relative to the ArUco tag's four corners

Camera 2 sees the same ArUco tag, finding its own position relative to the four corners of the same tag

Camera 1 and Camera 2 both know their positions in relation to a set of common points, so using linear regression we are able to estimate the transform between them.

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Our Code

We created a python script that obtains the images from the 2D cameras, calculates the 3D position of the ArUco tag relative to the camera, and returns this position in the form of a translation vector (x, y, z), and the axial rotations (roll, pitch, yaw).

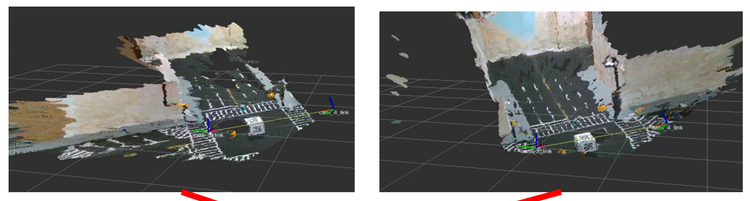
3D Models

To obtain 4 3D streams, we used Intel's Real sense D435 cameras. These have both a 2D color camera, and a stereo rgb-d camera.

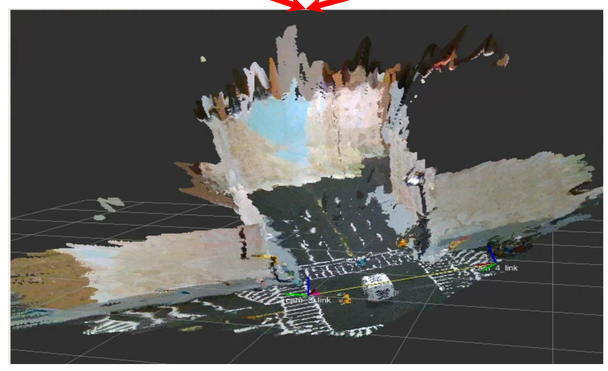
Stitching it all together

After we obtain the transform between the cameras we use the 'tf2_ros' Static Transformation Method to stitch them together, with the ArUco tag acting as the global center. 'tf2_ros' is a package for assigning transformations using Robot Operating System, a library with useful networking and 3D camera tools. Below is the output of one of the commands needed to apply the transform, and the effect viewed in Rviz.

```
cam_1
rosrun tf2_ros static_transform_publisher -0.1115381651148616
0.10903132958248486 1.3186237304527793 -2.4345006469276265 3.7
60841853800805 -0.039088643531957026 center cam_1_link
```



Camera 1 Camera 2



Conclusion

The benefits for a 3D stream of an intersection are still untapped. However, some realistic merits are

- Crash Prevention
- Pedestrian Detection
- Smart Car Location
- Multi-Car Communication

Our final script can be easily executed by users in order to recalibrate the cameras in the smart intersection whenever they are reset or moved.

WINLAB