In this document I will describe the high level design of the solution to my proposed project, “Tracking physical markers for rendering augmented reality objects.” Physical markers (also called fiducial markers) have a well defined size and pattern allowing them to serve as a reference point for location, rotation, and scale. There are several types of markers in use including ArUco, ARTag, AprilTag, Chilitags, and ARToolkit. I am choosing to implement recognition of AruCo markers as described in “Automatic generation and detection of highly reliable fiducial markers under occlusion” (S. Garrido-Jurado et al). ArUco has been implemented in multiple languages and environments and seems to have a good track record. The generation strategy for the markers emphasizes visual distinctiveness to decrease misidentification. AruCo’s square shape also allows for four reference points from each tracker by detecting the corners. Finally, the traditional high contrast black and white markers can be replaced by other color combinations (such as blue and green) for better detecting and masking occlusions.

The implementation of this system has a fairly straightforward computer vision pipeline as described in the paper, and the OpenCV implementation. First the image is converted to greyscale to better detect the outline of the markers. Then some sort of contour detection is applied to find the edges of the square markers. Polygon approximation selects rectangular features of interest, and discards the internal contours of the AruCo codes. The remaining features are normalized via perspective transformation. From this point, we can apply our detection algorithm and segment the image into a 2D matrix corresponding to the bit size of our marker. Next we look for the black cells that form the perimeter. If we find any white cells here we can discard the feature. Now that we are relatively certain the feature is a marker, we can read the bits stored and compare them to a pre known dictionary of markers. If there is a match, we will use the outline and location of the marker corners for future processing.

The other major component of this project is rendering virtual objects using the data retrieved from our markers. We can do some basic calculations to determine the center point of a marker, and its relative size and rotation. This will be the data used to orient our 3D model in space. I still need to do more work in researching how the markers can provide spatial location information, and the method for rendering objects into a processed image.

If I am able to successfully implement these two components, next steps could include occlusion masking to show real world objects on top of the virtual ones, interaction between virtual objects based on proximity, and interaction between physical and virtual objects.

For validating my solution throughout the development process, there are multiple ways I can obtain test data. As ArUco markers are fairly common, I could use existing databases to test detection in various environments. As the goal of this project is a simple form of augmented reality, the final solution should use a live feed from a camera to perform detection and render in real time. Testing could also be done using a live camera and physical trackers in different environments and lighting conditions.

While I have used OpenCV before, it's been a couple of years now and I anticipate I have a lot to catch up on and learn in order to implement this project. Important topics will include feature extraction, masking, perspective correction, and image subdivision and analysis.

https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html
https://cs-courses.mines.edu/csci507/schedule/24/ArUco.pdf