Lec6_transcript

Today, we're transitioning into the topic of perception. To quickly set the stage, last session we almost completed a manipulation system. This included our hardware abstraction, hardware simulation, and atop this, we constructed a differential inverse kinematics block. This block, after some integration, sent iiwa commands and positions. The differential block needed to know the current iiwa state, which we achieved through a vital line in our setup.

However, this entire setup hinged on an assumption represented here, where we had a gripper velocity trajectory coming in. This assumption formed a complete closed-loop system, which although functional internally, was based on a significant assumption: we knew where the red brick was located. Essentially, everything was built on the information of where exactly in the world I needed to reach, which isn't sufficient. At some point, we must use our sensors to determine the brick's location to perform the task effectively.

So far, we operated under the assumption of a 'perception Oracle'—an imaginary helper that could instantly tell us the location of objects in the world frame. We used cheat ports to directly inform us of the body pose of any object in our hardware station. Today, our goal is to stop relying on these cheat ports and start utilizing cameras to close the simplest loop around that whole system, enabling it to run on a real robot. I'm going to download a large MeshCat animation over my phone's 5G, which might burn my data plan, but it's necessary. This animation is large primarily because of the YCB objects it includes, such as a mustard bottle, which have very high-resolution texture maps.

These cameras I brought today aren't always with me, but they are essential for today's demonstration. They are RealSense cameras, models 415 and 435. They are amazing depth cameras that you can simply plug in via USB. I'll explain more about them as we proceed. What you see in the scene are these depth cameras strategically placed around, with three cameras per bin to understand why soon. The setup is almost the same as our last session, with the main difference being the inclusion of cameras in the scene.

We'll start by determining where the mustard bottle is using our perception system, then proceed with our standard gripper trajectory to move from point A to point B. Now, what you'll see getting left behind in this process is a point cloud. This cloud is computed by reading the cameras at Time Zero and performing some initial processing on the data to decide where to

grasp. At the moment of perception, I input these observations into the system, match them with a model of expected observations, and integrate all these during the lecture.

Today marks our first day delving into perception in this course, and we'll focus a lot on this topic going forward. The approach today is somewhat old-school—using geometric views of perception rather than just learning a deep network from images to whatever representation we want. Even though directly using deep networks from images is considered the most effective method today, understanding the fundamental geometry incorporated in many perception tools is crucial, whether they use neural networks or not. This knowledge forms the foundation necessary for working with advanced concepts like neural radiance fields or incorporating geometric priors into neural networks.

We will cover iterative closest point methods, discussing how it fits into today's perception strategies. This foundational understanding is essential for anyone interested in advancing their knowledge in fields like autonomous driving, where accurate perception is critical for identifying and navigating around pedestrians, or in augmented reality, where seamless integration of digital and real worlds relies on precise 3D reconstructions of the surroundings.