Drone Control Project Final Report- Team 2

DiMaggio Paris, Anuraag Thakur, Alaina Tulskie, Lambert Igene, Yue Pan

Data Collection:

The main purpose of creating this diverse dataset is to guarantee that anyone can successfully fly a drone with our system. At this stage, our dataset consists of 52,102 RGB images, which includes 5 unique classes of images, one for each type of gesture that will control the drone. We have closed-fist images to start the drone, images of an open hand to stop the drone, images of a thumbs-up to move the drone up, images of a thumbs-down to move the drone down, and images of a pointing finger to move the drone in a specified direction. We collected RGB images for each of the 5 gestures since we are training with the RGB images only and will use the depth in a later step in the pipeline. For example, we have closed-fist images to start the drone, images of an open hand to stop the drone, images of a thumbs-up to move the drone up, images of a thumbs-down to move the drone down, and images of a pointing finger to move the drone in a specified direction. Meanwhile, we also converted our dataset into a new dataset with 52,102 images, which embedded the key points generated by the Mediapipe solution, and we also have 20 videos of pointing hand gestures that stored relevant 3D data for further project needs.

To collect the images, we ran a script to collect 30 seconds of footage at a time for each gesture. We ran this script in 7 different locations with high, low, and medium lighting levels. Both the left and right hands were included in each image set. Four of our group members contributed to the dataset, and we also recruited volunteers of different genders and human races to collect the hand gestures. Moreover, the data collection is mainly around two lighting conditions(nature lighting, and medium lighting conditions) and two postures(sitting and standing poses) to fulfill the in-door circumstance requirements. Moreover, we estimated the best using distance for Intel RealSense Camera as 3 feet, and based on the distance and the in-door use purpose, we checked the whole dataset, and removed 31,546 images from the original dataset of 75,126 images due to distance and in-door lighting constraints. After re-collections based on the specific requirements(in-door lighting conditions, estimated distance, etc..), we maintained the dataset in 52,102 images as our final dataset.

In addition to this, we also converted our dataset into a new dataset with 52,102 images, which embedded the key points generated by the Mediapipe solution. Besides, we also stored the relevant 3D information videos by three different methods (based on the Grayscale, RGB coloring, and the camera RGB coloring) for further research needs.

Hence, our dataset includes both male and female hands and 3 different skin colors, and it can be regarded as a mature in-door RGB image human hand gesture dataset with 5 different gestures.

Hand Gesture Detection:

To classify the hand gestures, we first convert the RGB images to grayscale and pass them through a neural network. Then, to determine the best neural network architecture, we run a grid search algorithm combining nodes, number of layers, and activation functions. We selected 2,000 images from the dataset to speed up the algorithm and validated the model using 200 images. Once a good match was found, we trained the final model using the entire dataset. The testing was performed on a racially diverse dataset in all lighting environments and against various backgrounds, distances, and lighting conditions.

Pipeline:

The hand detection and gesture classification pipeline begins with the RGB-D camera stream. Next, we use MediaPipe to detect the presence of a hand in the image and obtain hand key points. Using these key points, we create a bounding box around the hand and input the cropped image into our trained model. The model returns a list of five integers representing the gestures, with the highest value indicating the predicted gesture. The drone takes off only when the closed gesture is detected for two seconds and moves in the direction predicted by the gesture. The drone lands when the open gesture is detected for two seconds.

Gesture Estimation:

To estimate the hand gestures, we use the fixed gesture detector. First, we input the 42 key points of the hand from MediaPipe into the network as an array list and process them to classify the gesture.

In addition to gesture detection, we implemented an algorithm to direct the drone in a specific direction based on the pointing gesture. If the model detects the pointing gesture in the color image, the algorithm uses the depth image to determine which direction the index finger is pointing. Using MediaPipe's hand key point detection, we extract the x and y pixel locations of the index finger knuckle and distal joint, which is close to the fingertip. The distal joint was chosen instead of the fingertip since it is less likely that the pixel at its key point will lie off the finger region and therefore provide incorrect depth values.

Then, using those X and Y pixel values, we save the depth value in the depth image at that pixel location, which represents the depth of the finger keypoint. The depth and color frames are aligned as they are read from the camera, so the images lie in the same coordinate space. We use only the X and Z (depth) values for the finger, since we already included two gestures for moving the drone vertically (thumbs down and thumbs up).

There are some instances where the depth value is 0 at the knuckle or distal joint locations. This appears in places on the image where the depth value could not be recorded properly and is therefore incorrect. If the depth at either the knuckle or the distal joint is 0, then we find the closest non-zero pixel to the 0 depth pixel using Manhattan distance. However, it is possible that the nearest non-zero point is off the finger. To prevent these cases from interfering with the results, we check to see if the difference in the depth values are reasonable, based on the average length of a person's hand. If the difference in depth is too large, then

When feasible and accurate depth values are determined, we convert the X and Z values into points in 3D space using the intrinsic camera parameters. Projecting them into 3D space allows the X, formally in pixel values, to be comparable with Z. We create a vector from the X and Z locations and convert into a unit vector.

The final step is to translate the directional vector into a command to send to the drone. The drone can only take a value with a magnitude 20 or above. We scale up the unit vector to have a length of 55 instead of 1, which allows us to understand the direction in a space that's applicable to the drone. If either the X or Z dimension has a magnitude below 20, that direction is discarded and the drone will fly 55 cm in one direction. Otherwise, the drone will be sent commands to fly in 2 directions using the values in the X and Z dimensions as input. If the X and/or Z values are positive, the drone will fly right and/or forward, respectively. If the X and/or Z values are negative, the drone is sent left and/or backwards.

In practice, the directions work well when the user is pointing directly to the left or right. The algorithm sometimes struggles to interpret a forward pointing figure, but it works more accurately if the user points their finger slightly down so the camera can see more clearly. Some of the diagonal pointing gestures are interpreted slightly differently than expected, but the algorithm will still choose the correct general directions. These issues tend to occur in rooms that are more brightly lit than others. This may be caused by lighting glare on the hand that interferes with the depth values. Overall, there are a few inconsistencies that occur depending on lighting, but the system most often will send the drone in the proper direction.