What is the exact setup of our bundle adjustment?

1. First we reorder the **19** images so that adjacent images have the most overlapping area.
2. For each pairs of neighboring images, we extract the keypoints and recover the R and T between the image pairs and use triangulation to recover the 3D points.
3. Then, two data structures are used to register the recovered 3D points and their corresponding 2D keypoints: a vector of vectors called "**2D points**" (named as "keypoints" in our code) and a vector called "**registered 3D points**".

- "**2D points**" (vector of vectors):

	keypoint1	keypoint2	keypoint3	keypoint4	keypoint5	keypoint6	keypoint7	keypoint8	keypoint9	keypoint10
Img1	-1	-1	-1	-1	-1	0	-1	-1	-1	-1
Img2	-1	-1	0	-1	-1	-1	-1	-1	-1	-1
Img3									-1	-1

In "**2D points**" table: Each element in a row represents a keypoint being detected in the image: if the value of an element is "-1", it means this keypoint is not matched to any points in other images, thus not contributing to any 3D points being reconstructed.

If the value of an element is not "-1": For example, "0"s in the rows of Img1 and Img2, it means that the keypoint6 of the Img1 and the keypoint3 of the Img2 matches each other and their 3D corresponding point is indexed as 0 in the "registered 3D points" vector below.

- "**registered 3D points**" (vector): vector of reconstructed 3D points.

idx	0	1	2	3	4	5	6	7	8	9
xyz location										

4. Every time when processing a new image, we update the "2D points" table and the "registered 3D points" table:

For example, when processing the Img3, if a keypoint in Img3 is matched with the keypoint3 of Img2, we would update the value for that keypoint in Img3 as 0, meaning, this keypoint matches an existing recovered 3D point indexed as 0 in "**registered 3D points**", thus we do not recover this keypoint in 3D space. Other than that, we append the newly recovered 3D points from the Img2 and Img3 to the "**registered 3D points**" table.

5. We enrich "2D points" and "registered 3D points" table, by processing Img1 and Img2, then Img2 and Img3, then Img3 and Img4….. we register both 2D points and 3D points recovered.

6. After "**2D points**" and "**registered 3D points**" table are established, we start the **Bundle Adjustment :**
We loop thru each image and (using "**2D points**" and "**registered 3D points**" table) find out the 3D points that are recorded by that image and the 2D locations of these 3D points, then we compute the re-projection errors and summed up over all images. In the end, Ceres optimize this error.