detect_lines: Just call cv2.Canny and cv2.HoughLinesP directly with given parameters.

get_pairwise_intersections: Use two for loop to get two pair of line. For each pair, compute the intersection point. After that, remove the item with $z = 0$.

get_support_mtx: Use two for loop to loop though each element of *support_mtx*. Calculate the distance between correspondence point and line. Then set the element to be 1 if the distance is smaller than the distance threshold.

get_vanishing_pts: Directly sum up the row vector of *support_mtx* and get the index of maximum row sum. That index indicate the vanishing point from list of intersections.

get_vanishing_line: From give two point $\boldsymbol{a}$ and $\boldsymbol{b}$, return $\boldsymbol{a} \times \boldsymbol{b}$

get_target_height:

1. Get $\boldsymbol{u} = (\boldsymbol{b}_2 \times \boldsymbol{b}_1) \times \boldsymbol{l}$, where $\boldsymbol{l}$ is horizontal vanishing line
2. $\boldsymbol{l}_2 = \boldsymbol{v} \times \boldsymbol{b}_1$
3. $\boldsymbol{t} = (\boldsymbol{t}_2 \times \boldsymbol{u}) \times \boldsymbol{l}_2$
4. Calculate the distance v, t_1, t which is the distance between point $\boldsymbol{b}_1$ and $\boldsymbol{v}$, $\boldsymbol{t}_1$, $\boldsymbol{t}$ respectively
5. $h = d_1 \cdot \dfrac{t}{t_1} \cdot \dfrac{v - t_1}{v - t}$