Computer Vision – HW 4

1. Assume that you apply the optical flow algorithm (OF) on a pair of images with a given set of parameters. Let p be a pixel for which the algorithm fails to compute the OF. Moreover, changing a single parameter results in computing the OF of p. Write what the parameter may be, why the algorithm fails in the first case and why it succeeds in the second case.

   Answer:

   When applying the OF algorithm to a pair of images with a specific set of parameters, and it fails to compute the OF for a particular pixel p, adjusting the **window size parameter** around p could resolve the issue. The OF algorithm operates under the assumption that every pixel within a designated patch around the central pixel moves with the same optical flow. The initial failure to compute the OF for p may stem from an incorrectly sized window.

   If the window is too small, it might result in the "white wall" effect, where movement is undetectable because the window does not capture enough texture or features to estimate motion (rank(C) = 0).

   Conversely, a window that is too large might encompass pixels that are not moving together, leading to inaccurate OF computation due to the inclusion of multiple motion patterns. By adjusting the window size, we can ensure it is optimal for capturing the uniform motion of the pixel p and its surroundings, thereby enabling successful OF computation.

2. Consider the optical flow algorithm which we learned in class. On which camera motion it is expected to fail? Give a short explanation to your answer.

   Answer:

   The OF algorithm is likely to encounter difficulties with rapid camera movements, encompassing both rotational and translational motions. This limitation stems from the algorithm's foundational assumption that movements between consecutive frames are minimal, enabling the use of the Taylor series expansion to approximate changes. Rapid camera movements disrupt this premise by causing significant shifts in pixel positions across frames, exceeding the small changes anticipated by the Taylor series approximation. Consequently, the optical flow algorithm struggles to maintain accuracy, as the large displacements invalidate its underlying mathematical model.

3. Assume two pixels have the same optical flow and the camera is static. Does it necessarily imply that they are projections of two 3D points that move at the same 3D direction? If so, explain why, and if not give a specific counter example.

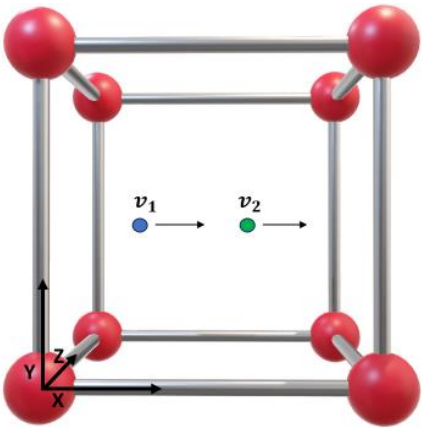
Answer:

No, identical optical flow for two pixels in a static camera's view does not guarantee that their corresponding 3D points are moving in the same real-world direction. Consider a scenario with a stationary camera capturing the motion of two points. In the 2D projection on the camera's sensor, both points might appear to move horizontally from left to right, exhibiting the same optical flow in the image plane.

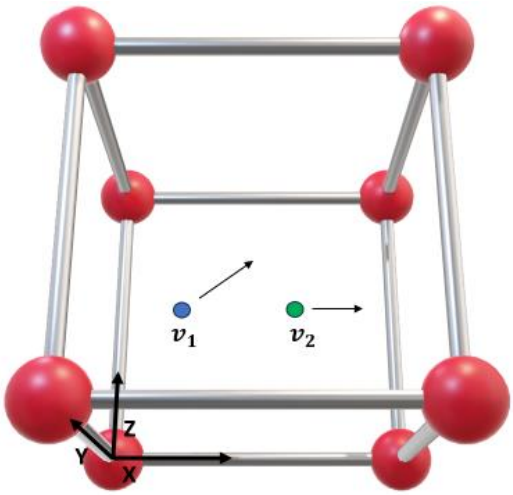
However, in the three-dimensional space, these two points could be moving along different trajectories. For instance, one point could be moving horizontally (v1), while the other could be moving both horizontally and away from the camera along the Z-axis (v2), but with a component of its velocity vector projecting onto the image plane in the same horizontal direction as the first point. Since optical flow only measures the apparent motion in the image plane, it captures the projection of the actual 3D motion vectors, not their full three-dimensional direction and magnitude.

Counter example:

2D Projection: The optical flow vectors (v1 and v2) in the image may appear identical onto the camera's image plane.



3D Reality: The actual motion vectors in 3D space could differ in both direction and speed. One point could indeed be moving directly across the field of view, while the other could be moving in a different direction, such as diagonally away in the 3D space.



4. Consider two images of the same static scene that were taken from two cameras. Assume that the COP of the cameras are identical (no translation only rotation and maybe different internal parameters).
   Prove formally: the two images are related by an homography transformation. Hint: You can prove it using the assumption that the world coordinate system is the same as the coordinate system of one of the cameras, and the rotation between the cameras, as well as the internal parameters of each camera are known. If you use these assumptions, you must explain why it is ok to use them.

   Answer:

   Let's assume that the world coordinate system is aligned with the coordinate system of Camera 1.

   Projection from Camera 1: The projection of the 3D point P onto the image plane of Camera 1 can be represented as $p_1 = K_1[I|0]P$, where K_1 is the matrix of internal parameters for Camera 1, I is the identity matrix, and 0 is the zero translation vector since the COP is at the origin of the world coordinate system.

   Let $P = (P_x, P_y, P_z, 1)$ be a point in the 3D real world, in homogeneous coordinate system.

   Let $M_1 = M_{int1}M_{ext1}$ be the projection matrix of camera 1

   $M_{int1} = K_1$, which is the matrix of internal parameters for Camera 1

   $M_{ext1} = [R_1|-R_1T_1] = [I|0]$, because the coordinate system of camera 1 is aligned with the real-world coordinate system, so the translation vector is 0, and the rotation matrix is the identity matrix

So, in overall, $M_1 = K_1[I|0] = \begin{pmatrix} s_{x1}f_1 & 0 & o_{x1} & 0 \\ 0 & s_{y1}f_1 & o_{y1} & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$

Now, $\tilde{p}_1 = M_1\tilde{P} = \begin{pmatrix} s_{x1}f_1 & 0 & o_{x1} & 0 \\ 0 & s_{y1}f_1 & o_{y1} & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ P_z \\ 1 \end{pmatrix} = \begin{pmatrix} s_{x1}f_1 & 0 & o_{x1} \\ 0 & s_{y1}f_1 & o_{y1} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix}$

So, we can basically remove the translation vector, and consider M_1 to be K_1

Thus, $p_1 = K_1P,\ where\ P = (P_x, P_y, P_z)$

Projection from Camera 2: For Camera 2, which is related to the first camera by a rotation R_2 (and no translation), the projection matrix is $M_1 = K_1[I|0]$, So, the projection of P onto its image plane can be represented as $p_2 = K_2[R_2|0]P$, where K_2 is the matrix of internal parameters for Camera 2.

Now, the same as before, $\tilde{p}_2 = M_2\tilde{P} =$

$$\begin{pmatrix} s_{x2}f_2 & 0 & o_{x2} & 0 \\ 0 & s_{y2}f_2 & o_{y2} & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} R_{11} & R_{12} & R_{13} & 0 \\ R_{21} & R_{22} & R_{23} & 0 \\ R_{31} & R_{32} & R_{33} & 0 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ P_z \\ 1 \end{pmatrix} =$$

$$\begin{pmatrix} s_{x2}f_2R_{11} + o_{x2}R_{31} & s_{x2}f_2R_{12} + o_{x2}R_{32} & s_{x2}f_2R_{13} + o_{x2}R_{33} \\ s_{y2}f_2R_{21} + o_{y2}R_{31} & s_{y2}f_2R_{22} + o_{y2}R_{32} & s_{y2}f_2R_{32} + o_{y2}R_{33} \\ R_{31} & R_{32} & R_{33} \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix}$$

Hence, $p_2 = K_2R_2P$

To find the relationship between p_1 and p_2, we can use the fact that both are projections of the same world point P. By isolating P from the first equation and, we get: $p_2 = K_2R_2K_1^{-1}p_1$. This is possible because K_1 is 3x3 matrix that has an inverse.

So, $\mathbf{H = K_2R_2K_1^{-1}}$ is the homography matrix that relates p_1 and p_2. Q.E.D

Assumptions:

- The assumption that the world coordinate system can be the same as the camera coordinate system is just a choice of reference. It simplifies the math without affecting the generality of the proof because the relationship between two images of a static scene is independent of the particular coordinate system chosen.
- It is ok to assume that we know the rotation matrices between the cameras and their internal parameters because in a standard stereo vision systems these parameters are necessary to establish the relationship between the

two image planes. If we don't know them, it would be impossible to calculate the H matrix and the projections of P.

5. Consider a video captured by a camera that is attached to the side of a car that moves on a straight road. Assume you have a perfect optical flow algorithm, which is applied to the video.
   What is the expected optical flow (OF) of the projection of the buildings that are parallel to the road with the same distance to the road? Describe its orientation and size, and whether it is fixed for all pixels that are projection of these buildings. Give a short explanation for your answer.

   Answer:

   In a video captured by a camera affixed to the side of a car, the OF of the projection of buildings that are parallel to the road and equidistant from it would display specific characteristics.

   Orientation of OF: The OF vectors would be horizontal in the image plane, given the camera's perpendicular orientation to the direction of motion. The OF vectors for these buildings would point in the direction opposite to the car's motion because the buildings would appear to move past the camera from the front to the back as the car advances.

   Size of OF: Regarding the size of the OF vectors, they would not be uniform across the entire projection of the buildings. Instead, the magnitude of the OF would vary depending on the position in the image:

   The OF magnitude would be larger for pixels of buildings that are closer to the principal point in the image (center of the image, perpendicular to the camera). Because their angular speed is higher.

   The OF magnitude would be smaller for pixels towards the edges of the image. Pixels of buildings at the right and left edges of the images, would have a smaller OF vectors, because their distance from the camera is larger and thus the angular speed is smaller.

6. Assume the OF we learned in class is applied once to a pair of successive frames and once to frames that are 20 frames apart using the same set of parameters. You may assume that the camera motion is constant, and that the scene is static.
   1. (a) On which regions the computation of the OF is expected to fail in both cases? Give a short explanation to your answer, including algebraic justification.
   2. (b) Where the OF is expected to fail only for the 20 frames apart case? Explain your answer and suggest a method to overcome this failure.

   Answer:

   (a) The regions where the Optical Flow (OF) computation is expected to fail in both cases are uniform or textureless regions. In such areas, the image intensity remains

constant, offering no distinctive features for the OF algorithm to track between frames. The algebraic justification is based on the OF equation: $I_y V_y + I_y V_y = -I_y$, where I_x and I_y are spatial gradients of the image intensity along the x and y axes, respectively, V_x and V_y are the velocity components in those directions, and I_t is the temporal gradient of image intensity. If I_x and I_y are zero, which occurs in uniform regions without intensity variation, the left side of the equation becomes zero. Since It may not be zero due to noise or illumination changes, we cannot solve for V_x and V_y because we essentially have an equation of the form $0 = -I_t$, which is unsolvable for V_x and V_y

(b) For the case of frames that are 20 frames apart, the OF computation is expected to fail in regions where objects are moving rapidly. This is because such movement causes large displacements that violate the small motion assumption fundamental to the OF calculation and the Taylor series expansion it employs. To address this, a multi-scale pyramid approach can be used. This technique involves constructing scaled-down versions of the original images and computing OF starting from the lowest resolution, gradually moving up to full resolution. This allows the algorithm to capture large movements at the coarsest scale, where they appear smaller, and iteratively refine the motion estimation as it proceeds to finer scales. By doing so, the displacement between frames becomes small enough at some level of the pyramid to satisfy the small motion assumption, thus enabling accurate OF computation.

7. The basic change detection algorithm we learned in class, computes a single image as a background model using a median image. The algorithm has several parameters.
    1. (a) List the set of parameters.

       n - how many frames to use for background calculation (median pixel value).

       th - Threshold for which we classify a non-background.

    2. (b) Assume that the intensity values of the pixel p at frame i is given by p[i], and p = [10,10,100,10,202,10,30,205,201,200,201]. Using one set of parameters, the value p(11) = 201 was detected as background while using another set of parameters, the value p(11) = 201 was detected as foreground. Give a single parameter of the algorithm such that changing its value can explain this difference. Give a short explanation for your answer including the list of all parameters in the two cases.

       The threshold parameter and the number of images that are included in the median calculation are heavily important to determine whether a pixel belongs to the background or not. Let's see an example:

       let th = 0.5 and let's assume the background is always computed for the last n frames.

If n=3 then $d_k(p) = |I_B(p) - I_K(p)| = |201 - 201| = 0$

$0 < 0.5$ so p(11) will be classified as background.

If n=5 then $d_k(p) = |I_B(p) - I_K(p)| = |200 - 201| = 1$

$1 > 0.5$ so p(11) will be classified as foreground.

8. Suggest a change detection algorithm where the background value of each pixel is based on a patch around the pixel rather than the intensity values of the pixel. List the set of parameters of your algorithm. Discuss the pros and cons of using a patch rather than a pixel to model the pixel background value.

Answer:

Suggested Algorithm:

1. calculate a median background model I_b using a set number of frames.
2. Calculate the mean intensity of the patch w(p) surrounding pixel p in the current image, denoted as $I_k(w(p))$.
3. Calculate the mean intensity of the patch w(p) surrounding pixel p in the background model, denoted as $I_b(w(p))$.
4. Compute the absolute intensity difference $d_k(w(p)) = | I_b(w(p)) - I_k(w(p)) |$ for each pixel.
5. If $d_k(w(p)) > th$, classify pixel p as part of a moving object.

Note that it is possible to calculate the mean intensity of a patch using a Gaussian filter, similar to the method used in Harris Corner detection. In this context, the standard deviation of the Gaussian filter (σ) effectively determines the size of the patch, dictating how many neighboring pixels are included when calculating the mean intensity.

The parameters of the algorithm are:

1. Th - Threshold for change detection.
2. N - Number of frames used to calculate the median background.
3. Patch size / σ (Sigma neighbors) – Determines the size of the patch around the pixel.

Pros and Cons of Using a Patch vs. Pixel for Background Modeling:

Pros:

- Increased Robustness: The approach is less sensitive to small changes in intensity, which can help in ignoring irrelevant details, like camera noise or changes in illumination that are not due to motion.

Cons:

- Potential Loss of Detail: By averaging over a patch, there is a potential loss of detail which could result in the algorithm missing subtle movements in pixel p.
- Increased Computational Load: Computing the mean intensity over a patch for every pixel can be computationally intensive, especially for larger patch sizes or higher-resolution images.

9. Assume that two images of the same object are captured by two cameras with the same internal parameters, but from a different distance. Assume that in order to match feature points, the Harris corner detector is applied to the two images. Which parameter of the corner detector should be modified in order to obtain corresponding corners? Give a short explanation to your answer.

   Answer:

   To match feature points between two images of the same object captured from different distances using the Harris corner detector, the parameter that needs to be adjusted is the size of the analysis window, often related to the Gaussian filter's sigma (σ) used in the algorithm. This parameter determines the scale at which the algorithm detects corners in the image.

   In the context of images captured from different distances, objects closer to the camera will project larger on the image sensor than those further away. Therefore, the scale of the detected features will vary between the two images. To match these features, the Harris corner detector must be applied at corresponding scales.

   Adjusting the window size or the sigma value allows the detector to identify features that are equivalent across scales. For the image captured from a closer distance, where features appear larger, a larger window size should be used to encompass the entire feature. Conversely, for the image captured from a further distance, a smaller window size is appropriate to match the scale of the features in the first image.

10. Assume that you are given the projections of the 3D points $\{P_i\}^k_{i=1}$ onto two images that were captured from different locations. Let $\{p_i\}^k_{i=1}$ and $\{q_i\}^k_{i=1}$ be the sets of these projections. Moreover, you are given that p_i and q_i are corresponding points and $k > 50$. Suggest a method to test whether $\{P_i\}^k_{i=1}$ are located on a single plane. If your method requires parameters, list them and explain how they affect the results.

    Answer:

    Assuming we have two sets of projected points $\{p_i\}$ and $\{q_i\}$ from the projections of 3D points $\{P_i\}$ onto two images taken from different locations, we can use the following method to test if the 3D points lie on a single plane:

Suggested Method:

1. **Calculate the Matrix H**: Use the RANSAC algorithm to estimate the homography matrix H that best aligns the point sets p_i with q_i.
2. **Apply H**: Transform the points p_i using matrix H to obtain the corresponding points p_i'.
3. **Error Calculation**: Measure the difference between the homography-transformed points p_i' and the corresponding points q_i.
4. **Thresholding the Error**: Establish a threshold value for the error. If the calculated distances for the majority of point pairs are below this threshold, it suggests that the original 3D points P_i reside on the same plane.

Parameters Required:

- **Error Threshold**: This parameter determines how closely the transformed points must match the corresponding points to be considered on the same plane. A lower threshold increases the precision but may discard inliers as outliers, whereas a higher threshold is more forgiving but might include outliers.
- **RANSAC Iterations**: The number of iterations affects the probability of finding a homography matrix that includes only inliers. More iterations increase the chance of a correct model.
- **RANSAC Inlier Threshold**: This threshold decides which points are considered inliers during the RANSAC process. A smaller value will result in a stricter model that could exclude valid points, while a larger value may incorporate outliers.