Agv Task

Task 1

1. Aw Shii, Here we flow again...

Subtask 1

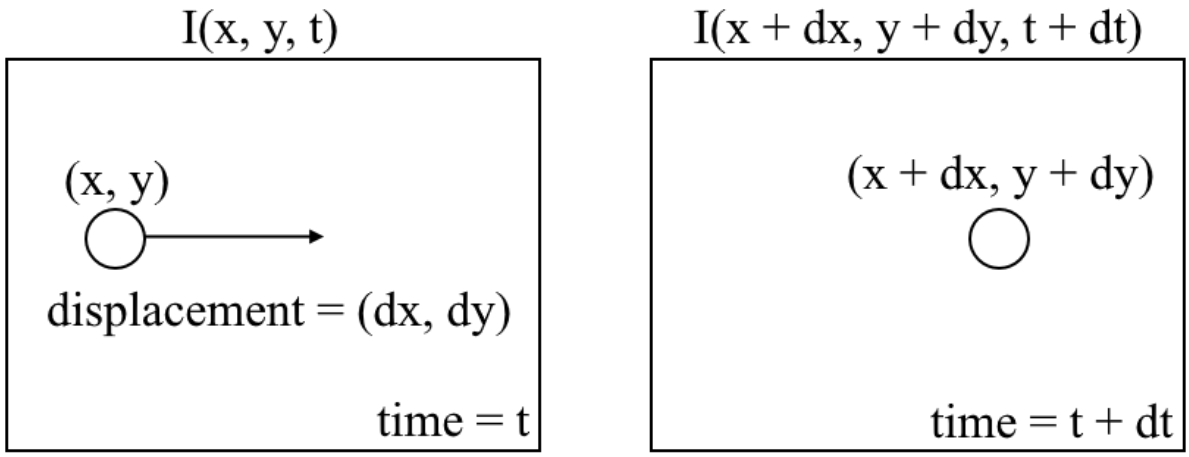
CJ NEEDS HELP!: IMPLEMENTING OPTICAL FLOW

What is Optical flow

Optical flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer and a scene.

Here in optical flow we use an intuitive assumption that there is no change in brightness even if a point moves between two frames .

From the above assumption we get some mathematical equations



Between consecutive frames we can express image intensity I as a function of x,y,t .

So when after dt time there will be dx and dy displacement respectively . First, we assume that pixel intensities of an object are constant between consecutive frames.

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t)$$

After applying taylor series approximation on the right hand side term we get

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + \dots$$

$$\Rightarrow \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t = 0$$

Hence after this we divide the equation by dt so that we get

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0$$

We notice that here we have two variable and a single equation so the get the answers we would require more assumption or more equations for that we use lucas-kanade method .

Difference between Dense and Sparse Optical flow

Sparse Optical Flow (e.g., Lucas-Kanade method) tracks only a subset of "interesting" pixels (like corners or edges). It is computationally efficient but might miss some motion details.

Dense Optical Flow (e.g., DeepFlow, or Gunnar Farneback's algorithm) estimates motion for all pixels in a frame. It provides more accurate motion estimation but is computationally expensive .

Implementing Sparse Optical Flow

Sparse optical flow selects a sparse feature set of pixels (e.g. interesting features such as edges and corners) to track its velocity vectors (motion). The extracted features are passed in the optical flow function from frame to frame to ensure that the same points are being tracked. There are various implementations of sparse optical flow, including the Lucas-Kanade method, the Horn-Schunck method, the Buxton–Buxton method, and more. We will be using the Lucas-Kanade method with OpenCV, an open source library of computer vision algorithms, for implementation.

Shi-Tomasi Corner Detector

For the implementation of sparse optical flow, we only track the motion of a feature set of pixels. Features in images are points of interest which present rich image content information. For example, such features may be points in the image that are invariant to translation, scale, rotation, and intensity changes such as corners.

1. Determine windows (small image patches) which have a rich texture and have high variations in intensity as we move along x and y .
2. For each window compute a score R .

3. Depending on the value of R the window is classified as flat , corner , or edge .

How to calculate R and what is R

For small shifts $[u,v]$ we have a *bilinear* approximation:

$$E(u,v) \cong [u,v] \quad M \quad \begin{bmatrix} u \\ v \end{bmatrix}$$

where M is a 2×2 matrix computed from image derivatives:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Shi-Tomasi Proposed the scoring function as : R = min(λ1 , λ2) This basically means if R is greater than the threshold than it is classified as corner or a good point to track for sparse optical flow . There is also a function in opencv `goodFeaturesToTrack()` which we will discuss later .

Lucas-Kanade Sparse Optical flow :

Lucas-Kanade method works by comparing to different frame

There few Assumptions we have to take in Lucas-Kanade method

1. Two time frames are separated by a small time increment (dt) such that objects are not displaced significantly
2. Second , we assume that the all the window patch or size moves in a same direction and they all have same motion

Now , we can take a small 3*3 window using the Shi-tomasi corner detector technique and we can represent the equations as

$$I_x(q_1)V_x + I_y(q_1)V_y = -I_t(q_1)$$

$$I_x(q_2)V_x + I_y(q_2)V_y = -I_t(q_2)$$

$$\vdots$$

$$I_x(q_n)V_x + I_y(q_n)V_y = -I_t(q_n)$$

Here q1 to qn are the pixels and The above set of equations can be represented as Av = b.

$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1) \\ I_x(q_2) & I_y(q_2) \\ \vdots & \vdots \\ I_x(q_n) & I_y(q_n) \end{bmatrix} \qquad v = \begin{bmatrix} V_x \\ V_y \end{bmatrix} \qquad b = \begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ \vdots \\ -I_t(q_n) \end{bmatrix}$$

Now we have 9 equations and two variables so it is overdetermined so we least square fitting to get the two variables

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i)I_y(q_i) \\ \sum_i I_y(q_i)I_x(q_i) & \sum_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i)I_t(q_i) \\ -\sum_i I_y(q_i)I_t(q_i) \end{bmatrix}$$

However we know we can apply lucas-kanade for small movements only so to track big movements we can use pyramid method . In basic terms we can understand this as we decrease the resolution the small movements become negligible and big movements become small and hence we can apply lucas-kanade method . There is a function in opencv for calculating lucas-kanade optical flow `calcOpticalFlowPyrLK()` we will discuss about this later .

Hence till here we have successfully understood the maths and logic behind use of optical flow using lucas-kanade optical flow method . As for understanding the code part it is well commented and we can understand all of it from there directly .

Subtask 2

GOING WITH THE FLOW

1. Gradient-Based Control Using Optical Flow
2. Target Navigation Using Artificial Potential Fields (APF)
3. Final Control Equations for AI2-THOR Agent Movement

We have already learned about the optical flow basics and lucas-kanade method for determining optical flow so we will not be discussing it again here .

Gradient-Based Control Using Optical Flow

From the computed motion vectors (Vx , Vy) we determine :

- **Dominant direction** of movement using mean flow or motion vectors :

  dx = mean(Vx) and dy = mean(Vy)

1. **Strategy for direction movement**

   If **dx is large and positive** - Motion is mostly in horizontal direction - Rotate **right**.

   If **dx is large and negative** - Motion is mostly in horizontal direction - Rotate **left.**

   If **dy is large and positive** - Motion is mostly in vertical direction - Move **forward .**

   If **dy is large and negative** - Motion is mostly in vertical direction - Move **Backward .**

Target Navigation Using Artificial Potential Field (APF)

Instead of just following the optical flow and moving randomly we use Artificial Potential field to move towards the target

Artificial Potential Field : It is a mathematical function where the target is a global minimum and they attract the bot towards it and the objects are maximum which repel the bot . Hence it moves according to these forces in the direction of the target .

$$F_{\text{att}} = \nabla U_{\text{att}} = \alpha \left(\sqrt{(x - x_{\text{goal}})^2 + (y - y_{\text{goal}})^2} \right)$$

Here (xgoal,ygoal) are the Target position and α is scaling factor for attraction force which we can change accordingly .

$$F_{\text{rep}} = \frac{\gamma}{d}$$

Here , gamma is the scaling factor and d is the distance between the bot and the closest obstacle .

After this Finally we calculate F which helps us in the determining the final direction for movement .

$$F = F_{\text{att}} + F_{\text{rep}}$$

If **|F_x| > |F_y|** than F_x will decide the movement and will in **Horizontal direction** and **vice-versa .**

If **F_x > 0** Then it will rotate **Right** and **vice-versa .**

If **F_y > 0** Then it will Move **Forward** and **vice-versa .**

Here dx and dy is used for immediate movement whereas F_x and F_y is used for Final target movement .

Logic Behind The Hybrid Navigation System

If **|dx| + |dy| > |F_x| + |F_y|** use **optical flow** or else use **Potential field forces .**