Assignment 6 Answers.

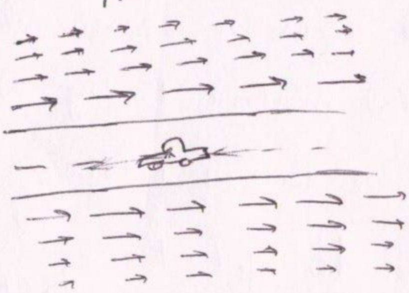
1. a) 3D motion vector define how the objects move in world.

2D projected motion vector are defined as the projection of 3D motion vector onto 2D, camera coordinates. We get these vector when we take picture or video of 3D objects.

2D observed motion, also called as optical flow, is what we observe in the image coordinates. These are a noisy estimate of the 2D projected motion vectors.

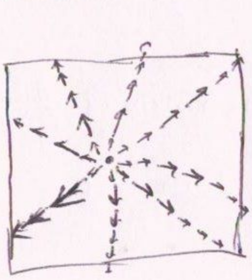
Yes, it is possible that motion in 3D will not produce optical flow vector. For example, the objects which are very far from the camera center, even though are moving, don't have/produce optical flow vectors.

b)



Projected motion vectors are larger at points closer to the car, and they decrease as the distance to the car increases.

c)



Projected motion vector are smaller closer to the place when plane is trying to land.

The projected motion vector around the point keep ~~increasing~~ increasing as distance increases, but decreases as the distance goes higher and higher.

d) $$v = \frac{f}{Z^2}(VZ - V_Z P)$$

$$V_Z = \frac{f}{Z^2}(V_Z Z - V_Z Z) = 0$$

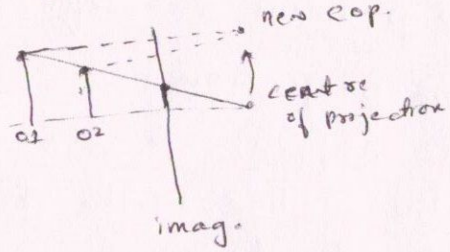
g) Equations for instantaneous epipole : $x_0 = \frac{T_x}{T_z} f$; $y_0 = \frac{T_y}{T_z} f$.

f: focal length ; (x_0, y_0) is the instantaneous epipole.

(T_x, T_y, T_z) is the Translational motion vector & its components.

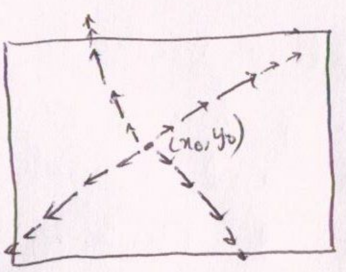
h) Motion parallax is created when 2 points at some point in time coincide on image and then they appear to move differently.

Motion parallax is defined as apparent motion of 2 instantaneously coincident points.



f) Projected motion fields when it is pure translational motion

Case 1: $T_z \neq 0$ (eg: plane landing)



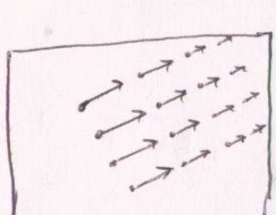
$$V_x = \frac{T_z}{Z}(x - x_0) \qquad V_y = \frac{T_z}{Z}(y - y_0)$$

$(x_0, y_0) \rightarrow$ instantaneous epipole.

Vectors far from the instantaneous epipole are bigger, but the higher the Z, lower is the magnitude of optical flow vector

Case 2: $T_z = 0$ (eg: driving car & looking out in side)

$$V_x = \frac{-T_x}{Z} f \qquad V_y = \frac{-T_y}{Z} f$$



- Vectors are parallel and move in same direction
- magnitude decreases as ~~[illegible]~~ farther
- instantaneous epipole is at infinity.

e) Translational motion vector:

$$V_x^{(t)} = \frac{T_z x - T_x f}{Z} = \frac{T_z (x - x_0)}{Z}$$

$$V_y^{(t)} = \frac{T_z y - T_y f}{Z} = \frac{T_z}{Z}(y - y_0)$$

Rotational motion:

$$V_x^{(\omega)} = -\omega_y f + \omega_z y + \frac{\omega_x x y}{f} - \frac{\omega_y x^2}{f}$$

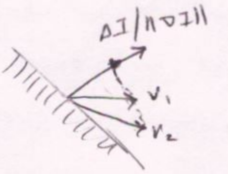
$$V_y^{(\omega)} = \omega_x f - \omega_z x - \frac{\omega_y x y}{f} + \frac{\omega_x y^2}{f}$$

2 a) $\frac{\partial}{\partial t}(I(x(t), y(t), t)) = 0$: Optical Flow Constraint Equation.

The basic assumption that is used to derive this equation is that "image brightness of object is constant throughout the image."

b) Aperture problem is defined as 2 motion vectors having the same projection.

When this happens we can only observe the motion in the direction of the gradient.



Based on a single point, we only hope to estimate motion in the direction of the gradient (or perpendicular vector).

c) Block-based optical flow estimation methods address the aperture problem by considering many points in the neighborhood of the current pixel, and averaging all the optical flows in the neighborhood. This give us the more smooth solution that satisfies the entire neighborhood.

d) objective : $\boxed{\sum_{(x,y)\in \text{Patch}} (\nabla I(x,y) \cdot V + I_t)^2 = E(v)}$

$v^* = \underset{v}{\operatorname{argmin}}\, E(v)$

Solution is given by $\nabla E(v) = 0$.

System of equations to be solved

$\frac{\partial}{\partial x_t} E = 0$ $\quad \frac{\partial}{\partial y_t} E = 0$

$\Rightarrow (I_x x_t + I_y y_t + I_t) I_x = 0$ and $(I_x x_t + I_y y_t + I_t) I_y = 0$

The purpose of weighted block method is to give importance/weights to the pixels close to the center rather than just having same weights for all pixels in the considered window.

Considering weight the objective changes to:

$E(v) = \sum_{(x,y)\in \text{Patch}} w(x,y)(I_x x_t + I_y y_t + I_t)^2$

$$w(x,y) = \frac{1}{\|(x,y) - \|x_c, y_c\| + 1}; \quad (x_c, y_c) \text{ is the centre.}$$

This leads to a better & improved solution, as we are giving more importance to the optical flow vectors that are closer.

$$\text{Solution} = \begin{bmatrix} \sum w I_x^2 & \sum w I_x I_y \\ \sum w I_x I_y & \sum w I_y^2 \end{bmatrix} \begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} -\sum w I_x I_t \\ -\sum w I_y I_t \end{bmatrix}$$

e) Affine motion model, rather than considering the optical flow vectors to be constant in the window, computer the optical flow vector at each point in the window which give more accurate solution.

Objective:

$$E(a) = \sum_{(x,y) \in Patch} \left(I_x(a_1 + a_2 x + a_3 y) + I_y(a_4 + a_5 x + a_6 y) + I_t\right)^2$$

Solution is given by $\nabla E(a) = 0$. (full equations are given at the end)

Once we find the model parameters, a_1, a_2, a_3, a_4, a_5 & a_6, we can recover motion vectors as.

$$v(x,a) = \begin{bmatrix} a_1 + a_2 x + a_3 y \\ a_4 + a_5 x + a_6 y \end{bmatrix}$$

f) Objective for global motion estimation (Horn - Schunck)

$$E(v(x,y,t)) = \int_{\text{image}} \underbrace{E_{OF}^2(v(x,y,t))}_{\text{optical flow}} + \underbrace{\alpha^2}_{\substack{\text{Smoothing importance} \\ \text{(user selectable)}}} \underbrace{E_S^2(v(x,y,t))}_{\text{Smoothness}}$$

the advantage of this solution is that we can define how much to smooth the optical flow vectors.

$$E_{OF}^2 = (I_x \cdot u + I_y \cdot v + I_t)^2 \; ; \; E_S^2 = U_x^2 + U_y^2 + V_x^2 + V_y^2$$

U: image formed by x-components of optical flow vector

V: " " y- " " "

- Estimating U & V is difficult and we have to take iterative approach
- Using this method, We are regularizing the optical flow vectors and removing out irregularities / errors. Advantage is that we are calculating all optical flows at a time; thereby using the fact that optical flows vectors in nearby windows are close to be similar

g) Horn-Schunck iterative global optical flow estimation algorithm works as below.

- Start with initial guess for u, v
- then we iterate to refine u, v as below.

$$u^{n+1} = \bar{u}^{(n)} - \frac{(I_x \bar{u}^{(n)} + I_y \bar{v}^{(n)} + I_t) I_x}{I_x^2 + I_y^2 + \alpha^2}$$

$$v^{n+1} = \bar{v}^{(n)} - \frac{(I_x \bar{u}^{(n)} \; I_y \bar{v}^{(n)} + I_t) I_y}{I_x^2 + I_y^2 + \alpha^2}$$

- and stop when there is not much change in ~~(u,v)~~ u & v

$$\max(\|u^{n+1} - u^n\|, \|v^{n+1} - v^n\|) < \tau$$

We can use Lukas-Kanade or affine flow motion estimation methods to find values to ~~be~~ initialize u & v the very first iteration of the algorithm.

2 e) (Continued)

Solution is given by:

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} = \begin{bmatrix} \Sigma I_x^2 & \Sigma I_x^2 x & \Sigma I_x^2 y & \Sigma I_x I_y & \Sigma I_x I_y x & \Sigma I_x I_y y \\ \Sigma I_x^2 x & \Sigma I_x^2 x^2 & \Sigma I_x^2 xy & \Sigma I_x I_y x & \Sigma I_x I_y x^2 & \Sigma I_x I_y xy \\ \Sigma I_x^2 y & \Sigma I_x^2 xy & \Sigma I_x^2 y^2 & \Sigma I_x I_y y & \Sigma I_x I_y xy & \Sigma I_x I_y y^2 \\ \Sigma I_x I_y & \Sigma I_x I_y x & \Sigma I_x I_y y & \Sigma I_y^2 & \Sigma I_y^2 x & \Sigma I_y^2 y \\ \Sigma I_x I_y x & \Sigma I_x I_y x^2 & \Sigma I_x I_y xy & \Sigma I_y^2 x & \Sigma I_y^2 x^2 & \Sigma I_y^2 xy \\ \Sigma I_x I_y y & \Sigma I_x I_y xy & \Sigma I_x I_y y^2 & \Sigma I_y^2 y & \Sigma I_y^2 xy & \Sigma I_y^2 y^2 \end{bmatrix}^{-1} \begin{bmatrix} -\Sigma I_x I_t \\ -I_x I_t x \\ -I_x I_t y \\ -I_y I_t \\ -I_y I_t x \\ -I_y I_t y \end{bmatrix}$$