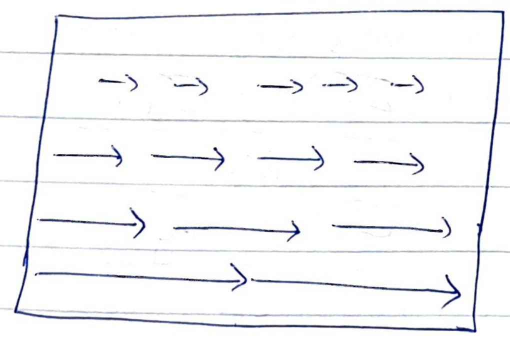
Assignment - 7

* Name = Parthkumar Patel
* Student ID = A20416508
* Course Number = CS-512
* Semester = Fall-2018.

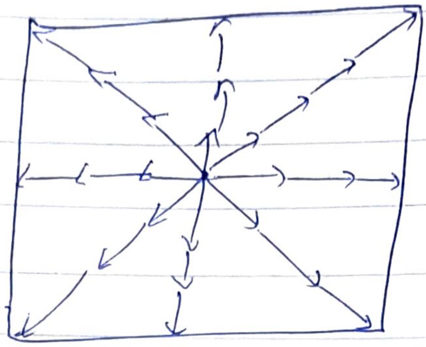
Que-1

(a) 3D motion vector is in the real world.
→ 2D motion vector is the projection of the 3D motion vector.
→ Optical flow is what we observed ~~2D motion vectors~~. in the image coordination
→ Yes, it is possible that motion in 3D will not produce optical flow vector.

(b) The objects close to camera will be bigger. the objects away from camera will be smaller.
3D motion is projected to the camera



(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 as distance increases, but decreases as the distance goes higher and higher.

(d)

$$V = \frac{f}{Z^2}(VZ - V_z P)$$

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

(e)

Translational motion vector =

$$V_x(z) = \frac{Z_z x - Z_x f}{Z} = \frac{Z_z}{Z}(x - x_0)$$

$$V_y(z) = \frac{Z_z y - Z_y f}{Z} = \frac{Z_z}{Z}(y - y_0)$$

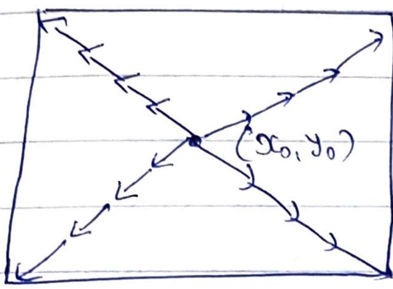
Rotational Motion =

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

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

(4)
→ Projected motion field when it is pure translational motion.

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

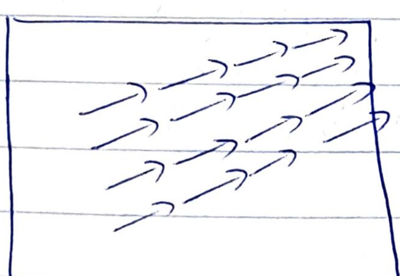


$$V_x = \frac{T_z}{Z}(x - x_0)$$

$$V_y = \frac{T_z}{Z}(y - y_0).$$

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



$$V_x = \frac{-T_x f}{Z},$$

$$V_y = \frac{-T_y f}{Z}.$$

→ Vectors are parallel and move in same direction
→ magnitude decreases as farther.
→ instantaneous epipole is at infinity.

(g) Equation for instanteneous epipole:

$$x_0 = \frac{\tau_x}{\tau_z} f \quad ; \quad y_0 = \frac{\tau_y}{\tau_z} f$$

f = focal length.
(x_0, y_0) = instanteneous epipole.
(τ_x, τ_y, τ_z) = Translational motion vector and its components

(h)
→ Motion parallax is created when 2 points at some point in time coincide on image and then appears to move differently.
→ Motion parallax is defined as apparent motion of 2 instanteneously coincided points.

relative motion field equations.

$$\Delta V_x = V_x - \overline{V_x}$$
$$\Delta V_y = V_y - \overline{V_y}$$

→ $(V_x, V_y, \overline{V_x}, \overline{V_y})$ get from =

$$\begin{cases} V_x = V_x(\tau) + V_y(\omega) \\ V_y \quad V_y(\tau) + V_y(\omega) \end{cases}$$

$$\begin{cases} \overline{V_x} = \overline{V_x}(\tau) + V_y(\omega) \\ \overline{V_y} = \overline{V_y}(\tau) + V_y(\omega) \end{cases}$$

Que-2

(a)
→ optical flow constraint equation.

$$= \frac{\partial}{\partial t}\left(I\left(x(t), y(t), t\right)\right) = 0.$$

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

(b)
→ Apeture 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 apeture problem by considering many points in the neighbourhood of the current pixel and averaging all the optical flows in the neighbourhood. This gives us the more smooth solution that satisfies the entire neighbourhood.

શ્રી સ્વામિનારાયણ ડિઝાઈન મિશન
Dt: / / Pg.no.:

(d)

$$\text{Objective} = \sum_{(x,y)\in \text{patch}} \left(\nabla I(x,y)\cdot V + I_t\right)^2 = E(V)$$

$$V^* = \underset{V}{\operatorname{argmin}}\ E(V) \Rightarrow \nabla E(V) = 0$$

→ System of equations to be solved.

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

$$\Rightarrow (I_x x_t + I_y y_t + I_t) I_x = 0 \quad \text{and}$$

$$(I_x x_t + I_y y_t + I_t) I_y = 0.$$

The purpose of weighted block methods 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 weights the objective changes

$$E(V) = \sum_{(x,y)\in \text{patch}} W(x,y)\left(I_x x_t + I_y y_t + I_t\right)^2.$$

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

→ This leads to a better and 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, computes the optical flow vectors 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 $\nabla E(a) = 0$.

→ Once we find the model parameters a_1, a_2, a_3, a_4, a_5 and a_6 we can recover motion vector 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)

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

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

→ Advantage is that we are calculating all optical flows at a time.

→ Another advantage is ~~the~~ all the regularization.

→ Estimating U & V is difficult and we have to take iterative approach.

(g)

→ Horn-Schunk iterative global optical flow estimation algorithm works as below.

(1) Start with intial guess for U, V.
(2) then we iterate to refine U, V as below:

$$U^{n+1} = \bar{U}(n) - \frac{(I_x \bar{U}(x) + I_y \bar{V}(n) + I_t) I_x}{I_x^2 + I_y^2 + \alpha^2}$$

$$V^{n+1} = \bar{V}(n) - \frac{(I_x \bar{U}^{(n)} . I_y \bar{V}^{(n)} + I_t) I_y}{I_x^2 + I_y^2 + \alpha^2}.$$

(3) Stop when there is not much change in U. and V.

$$\max(\| U^{n+1} - U^n \|, \| V^{n+1} - V^n \|) < \tau.$$

we can use Lucas-kanade or affine flow motion estimation methed to find values to intialize U & V the very first iteration of the algorithm.