



# **The 2D/3D Differential Optical Flow**

**Prof. John Barron**

**Dept. of Computer Science**

**University of Western Ontario**

**London, Ontario, Canada, N6A 5B7**

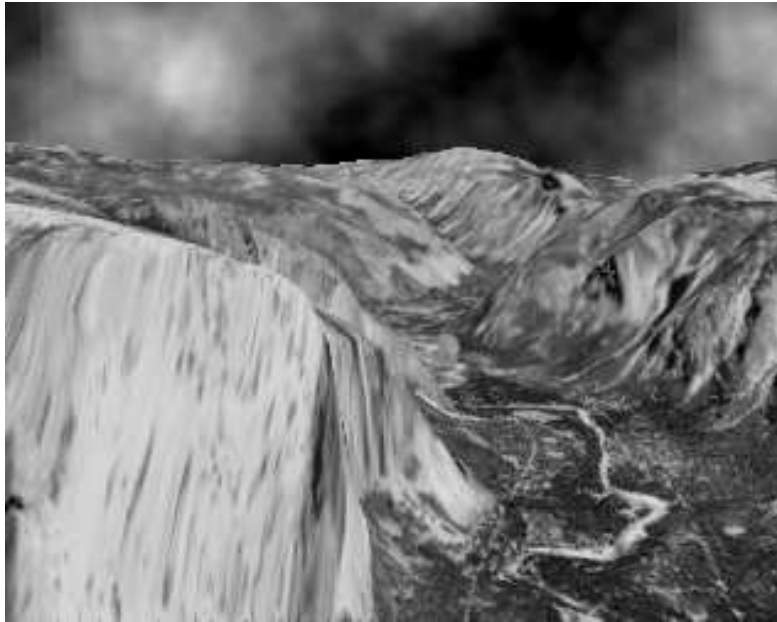
Email: [barron@csd.uwo.ca](mailto:barron@csd.uwo.ca)

Phone: 519-661-2111 x86896

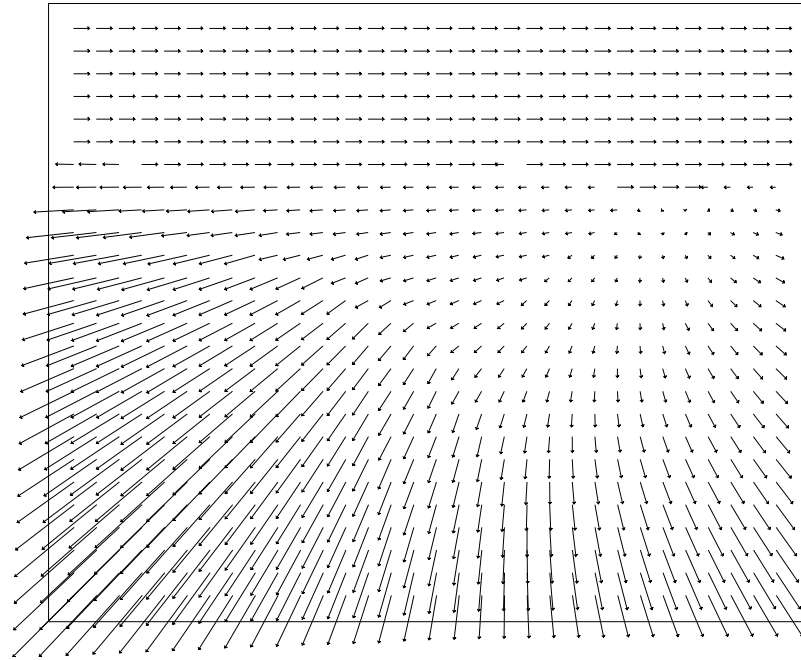
**Canadian Conference on Computer and Robot Vision (CRV2009)**

**Kelowna, British Columbia, May 24<sup>th</sup>, 2009**

## 2D Optical Flow: An Example



(a)



(b)

Figure 1: (a) The middle frame from the **Yosemite Fly-Through** sequence and (b) its correct flow field.

## 2D Optical Flow: An Overview

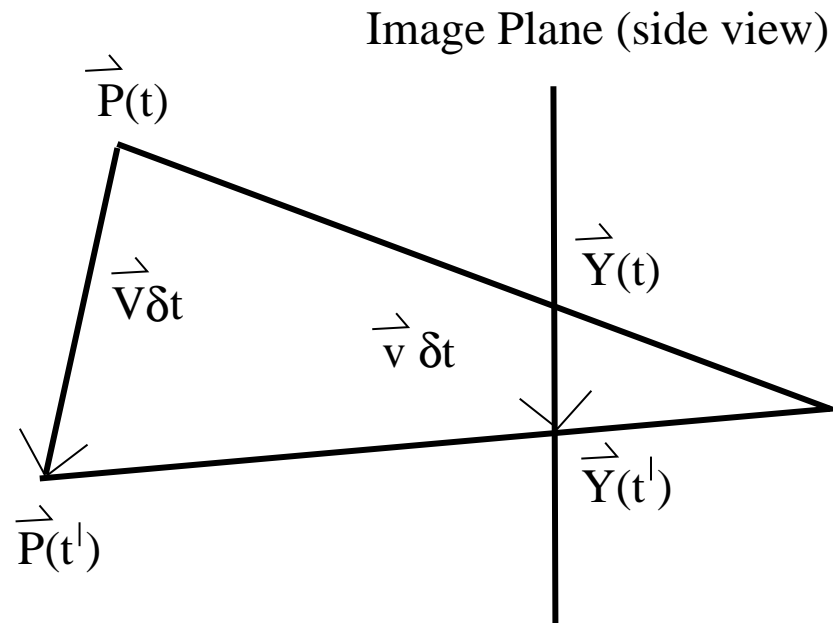


Figure 2:  $\vec{V}$  is the 3D velocity of 3D point  $\vec{P}(t) = (X, Y, Z)$ .  $\vec{v} = (u, v)$  is the 2D image of  $\vec{V}$ , i.e.  $\vec{v}$  is the perspective projection of  $\vec{V}$ . If  $\vec{P}(t)$  moves with displacement  $\vec{V}\delta t$  to  $\vec{P}(t')$  from time  $t$  to time  $t'$ , then its image  $\vec{Y}(t) = (x, y, f)$  moves with displacement  $\vec{v}\delta t$  to  $\vec{Y}(t') = (x', y', f)$  from times  $t$  to  $t'$ .  $f$  is the sensor focal.  $\vec{v}$  is known as **image velocity** or **optical flow**.

# The Image Velocity Equations

- The 3D instantaneous velocity  $\vec{V} = (U, V, W)$  of a 3D point  $\vec{P} = (X, Y, Z)$ , where the sensor moves relative to  $\vec{P}$  is

$$\vec{V} = (U, V, W) = -\vec{T} - \vec{\omega} \times \vec{P}, \quad (1)$$

where  $\vec{T} = (T_0, T_1, T_2)$  is the instantaneous sensor translation and  $\vec{\omega} = (\omega_0, \omega_1, \omega_2)$  is the sensor's instantaneous rotation. The components of  $\vec{V}$  can be written out in full as:

$$U = -T_0 - \omega_1 Z + \omega_2 Y \quad (2)$$

$$V = -T_1 - \omega_2 X + \omega_0 Z \quad (3)$$

$$W = -T_2 - \omega_0 Y + \omega_1 X \quad (4)$$

For a rigid object under perspective projection we can write

$$x = f \frac{X}{Z}, \quad y = f \frac{Y}{Z} \quad \text{and} \quad z = f \frac{Z}{Z} = f, \quad (5)$$

where  $f$  is the focal length of the sensor. The time derivatives of  $(x, y, z)$ ,  $(\dot{x}, \dot{y}, \dot{z}) = (u, v, 0)$ , yield the two non-zero components of instantaneous image velocity, which can be written as

$$\dot{x} = u = f \left( \frac{\dot{X}}{Z} - \frac{X\dot{Z}}{Z^2} \right) \quad (6)$$

$$\dot{y} = v = f \left( \frac{\dot{Y}}{Z} - \frac{Y\dot{Z}}{Z^2} \right) \quad (7)$$

Substituting equations (2), (3) and (4) into equations (6) and (7) and using the definition of perspective projection in equation (5) we obtain the standard image velocity equations (see Longuet-Higgins and Prazdny, Proc. R. Soc. London B208, 1981):

$$u = \frac{1}{Z} (-fT_0 + xT_2) + \omega_0 \left( \frac{xy}{f} \right) - \omega_1 \left( f + \frac{x^2}{f} \right) + \omega_2 y$$

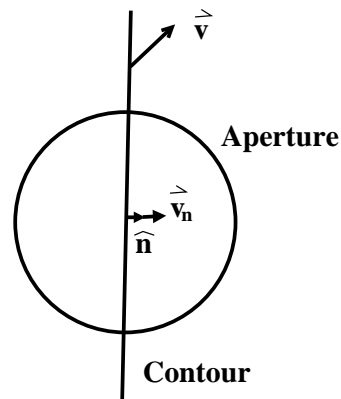
and

$$v = \frac{1}{Z} (-fT_1 + yT_2) + \omega_0 \left( f + \frac{y^2}{f} \right) - \omega_1 \left( \frac{xy}{f} \right) - \omega_2 x$$

- Given the sensor's instantaneous 3D translation and 3D rotation plus an image point's 2D coordinates  $(x, y)$  and its 3D depth,  $Z$  [hence we know  $X$  and  $Y$  as well via equations (5)] we can specify the correct 2D image velocity,  $(u, v)$ , for this image point.
- Conversely, if we can measure  $(u, v)$  values at image points, we can recover the 3D sensor translation scaled by 3D depth and 3D sensor rotation. We can also recover “relative” depth at image points in a scene. So, while we can't know from a single monocular sensor that object 1 is 4m away while object 2 is 8m away (absolute depth) we can tell that object 1 is twice as close as object 2 (relative depth). Of course, with a binocular sensor setup we can recover absolute motion and depth parameters.

# The 2D Aperture Problem

- We can estimate optical flow locally from spatial and temporal image intensity derivatives measured in local neighbourhoods.
- The aperture problem (Marr and Ullman 1981) is relevant when image velocities are measured locally.



**Figure 3:**  $\hat{n}$  is the unit normal vector in the normal velocity direction, i.e. it is perpendicular to the local contour structure and having length 1. Thus normal velocity  $\vec{v}_n$  is the component of full velocity  $v$  projected in the normal direction  $\hat{n}$ ,  $(\vec{v} \cdot \hat{n})\hat{n} = \vec{v}_n$ .

# The Significance of the Aperture Problem

- So, due to the aperture problem, we can measure just  $\vec{v}_n$  and not  $\vec{v}_t$  (the tangential velocity) or  $\vec{v}$  (the full velocity) locally.
- The aperture problem does not depend on the size of the aperture but rather on local contour structure: Is there enough local structure to recover full image velocity?

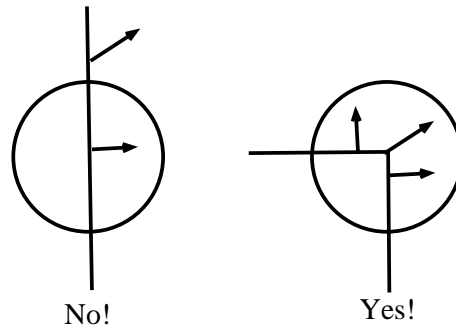
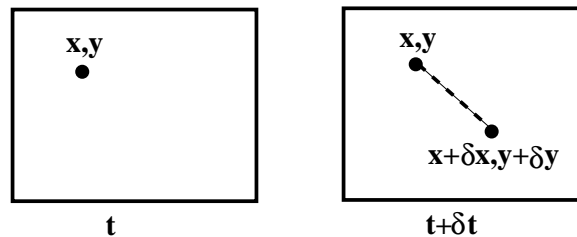


Figure 4: We can recover full image velocity in neighbourhoods of corner points but not in neighbourhoods when the local contour is straight.

# The Motion Constraint Equation

- Assume  $I(x, y, t)$  moves by  $\delta x$ ,  $\delta y$  in time  $\delta t$  to  $I(x + \delta x, y + \delta y, t + \delta t)$ .



- Since  $I(x, y, t)$  and  $I(x + \delta x, y + \delta y, t + \delta t)$  are the images of the same point:

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

- We can perform a 1<sup>st</sup> order Taylor series expansion about  $I(x, y, t)$ :

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) + \frac{\partial I}{\partial x} \delta x + \frac{\partial I}{\partial y} \delta y + \frac{\partial I}{\partial t} \delta t + H.O.T.$$

## The Motion Constraint Equation Continued:

- Because  $I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t)$  we obtain:

$$\begin{aligned}\frac{\partial I}{\partial x} \delta x + \frac{\partial I}{\partial y} \delta y + \frac{\partial I}{\partial t} \delta t &= 0, \\ \frac{\partial I}{\partial x} \frac{\delta x}{\delta t} + \frac{\partial I}{\partial y} \frac{\delta y}{\delta t} + \frac{\partial I}{\partial t} \underbrace{\frac{\delta t}{\delta t}}_{=1} &= 0, \\ I_x u + I_y v + I_t &= 0 \text{ and} \\ \nabla I \cdot \vec{v} + I_t &= 0.\end{aligned}$$

- Here  $u = \frac{\delta x}{\delta t}$  and  $v = \frac{\delta y}{\delta t}$  are the  $x$  and  $y$  components of image velocity and  $I_x = \frac{\partial I}{\partial x}$ ,  $I_y = \frac{\partial I}{\partial y}$  and  $I_t = \frac{\partial I}{\partial t}$  are image intensity derivatives at  $I(x, y, t)$ .

# Motion Constraint Line

- $\nabla I \cdot \vec{v} + I_t = 0$  is 1 equation in 2 unknowns (a line), the correct velocity is some unknown point on this line. The velocity with the smallest magnitude is the normal velocity  $\vec{v}_n$ .

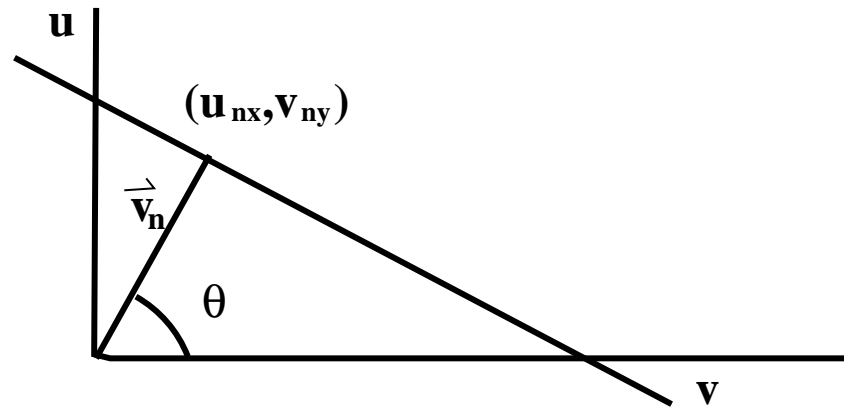
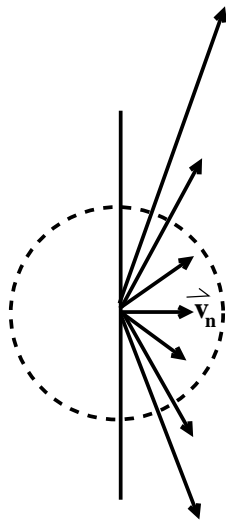


Figure 5: The motion constraint line.  $\vec{v}_n = (u_{nx}, v_{ny})$  is the velocity with the smallest magnitude that is on this line.

# The Relationship between Normal Velocity and the Motion Constraint Equation

- Consider a straight contour moving up/down and to the right with true full velocity  $\vec{v}$ . Since it is viewed through an aperture we can only see the motion perpendicular to the contour.



- Since the normal velocity is the smallest of all potential velocities it

is the point on the motion constraint line closest to the origin.

- We can compute  $v_n$  (the magnitude of  $\vec{v}_n$ ) and  $\hat{n}$  (its direction) and hence, the normal velocity,  $\vec{v}_n = v_n \hat{n}$ , very simply using the motion constraint equation  $\nabla I \cdot \vec{v} = -I_t$  and the fact that  $\vec{v} \cdot \hat{n} = v_n$  as:

$$\begin{aligned}\nabla I \cdot \vec{v} &= -I_t \\ \frac{\nabla I}{\|\nabla I\|_2} \cdot \vec{v} &= -\frac{I_t}{\|\nabla I\|_2} \\ \hat{n} \cdot \vec{v} &= v_n\end{aligned}$$

$$\Rightarrow \hat{n} = \frac{\nabla I}{\|\nabla I\|_2} \quad \text{and} \quad v_n = \frac{-I_t}{\|\nabla I\|_2}.$$

## Resolving the Aperture Problem

- We need to impose additional constraints to recover the full velocity  $\vec{v}$  everywhere.
- We could assume that each local image neighbourhood has constant velocity [Lucas and Kanade IJCAI1981]. Then 2 or more different normal velocities yield the true full velocity (in the least squares sense).
- We could assume that velocity varies smoothly everywhere [Horn and Schunck AI1981] and regularize a smoothness term across the image.

## 2D Lucas and Kanade 1981

- Given  $I_x$ ,  $I_y$  and  $I_t$  at a single pixel, we can compute  $\vec{v}_n$  as  $\vec{v}_n = \vec{v} \cdot \hat{n}$ , where  $v_n = \frac{-I_t}{\|\nabla I\|_2}$  and  $\hat{n} = \frac{\nabla I}{\|\nabla I\|_2}$  are as before.
- Given an  $k = n \times n$  neighbourhood with the same velocity  $\vec{v}$  we can write

$$\underbrace{\begin{bmatrix} n_{x1} & n_{y1} \\ n_{x2} & n_{y2} \\ \vdots & \vdots \\ n_{xk} & n_{yk} \end{bmatrix}}_N \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\vec{v}} = \underbrace{\begin{bmatrix} v_{n1} \\ v_{n2} \\ \vdots \\ v_{nk} \end{bmatrix}}_B$$

which we can write as  $\underbrace{N}_{k \times 2} \underbrace{\vec{v}}_{2 \times 1} = \underbrace{B}_{k \times 1}$ .

- For  $k \geq 2$  we can solve this system as  $\vec{v} = (N^T N)^{-1} N^T B$ .

## 2D Horn and Schunck 1981

- Horn and Schunck combined the motion constraint equation with a global smoothness term to constrain the estimated velocity field  $\vec{v} = (u, v)$ , minimizing:

$$f = \int_D (\nabla I \cdot \vec{v} + I_t)^2 + \lambda^2 (u_x^2 + u_y^2 + v_x^2 + v_y^2) dx dy$$

defined over a domain  $D$  (the image), where the magnitude of  $\lambda$  reflects the relative influence of the smoothness term.

- We use the Euler-Lagrange equations:

$$f_u - \frac{df_{u_x}}{dx} - \frac{df_{u_y}}{dy} = 0$$

$$f_v - \frac{df_{v_x}}{dx} - \frac{df_{v_y}}{dy} = 0,$$

where:

$$f_u = 2I_x(I_x u + I_y v + I_t)$$

$$f_v = 2I_y(I_x u + I_y v + I_t)$$

$$f_{u_x} = 2\alpha^2 u_x$$

$$f_{u_y} = 2\alpha^2 u_y$$

$$f_{v_x} = 2\alpha^2 v_x$$

$$f_{v_y} = 2\alpha^2 v_y$$

$$\frac{df_{u_x}}{dx} = 2\alpha^2 u_{xx}$$

$$\frac{df_{u_y}}{dy} = 2\alpha^2 u_{yy}$$

$$\frac{df_{v_x}}{dx} = 2\alpha^2 v_{xx}$$

$$\frac{df_{v_y}}{dy} = 2\alpha^2 v_{yy}.$$

- Since  $\nabla^2 u = u_{xx} + u_{yy}$  and  $\nabla^2 v = v_{xx} + v_{yy}$  we can rewrite the Euler-Lagrange equations as:

$$\begin{aligned} I_x^2 u + I_x I_y v + I_x I_t &= \alpha^2 \nabla^2 u \\ I_x I_y u + I_y^2 v + I_y I_t &= \alpha^2 \nabla^2 v. \end{aligned}$$

Using  $\nabla^2 u \approx \bar{u} - u$  and  $\nabla^2 v \approx \bar{v} - v$  we get

$$\begin{aligned} (\alpha^2 + I_x^2)u + I_x I_y v &= (\alpha^2 \bar{u} - I_x I_t) \\ I_x I_y u + (\alpha^2 + I_y^2)v &= (\alpha^2 \bar{v} - I_y I_t), \end{aligned}$$

- We can solve for  $u$  and  $v$  as:

$$\begin{aligned} (\alpha^2 + I_x^2 + I_y^2)u &= (\alpha^2 + I_y^2)\bar{u} - I_x I_y \bar{v} - I_x I_t \\ (\alpha^2 + I_x^2 + I_y^2)v &= -I_x I_y \bar{u} + (\alpha^2 + I_x^2)\bar{v} - I_y I_t, \end{aligned}$$

which can be written as:

$$\begin{aligned}(\alpha^2 + I_x^2 + I_y^2)(u - \bar{u}) &= -I_x [I_x \bar{u} + I_y \bar{v} + I_t] \\(\alpha^2 + I_x^2 + I_y^2)(v - \bar{v}) &= -I_y [I_x \bar{u} + I_y \bar{v} + I_t].\end{aligned}$$

- Gauss-Seidel iterative equations that minimize these equations are:

$$\begin{aligned}u^{k+1} &= \bar{u}^k - \frac{I_x [I_x \bar{u}^k + I_y \bar{v}^k + I_t]}{\alpha^2 + I_x^2 + I_y^2} \quad \text{and} \\v^{k+1} &= \bar{v}^k - \frac{I_y [I_x \bar{u}^k + I_y \bar{v}^k + I_t]}{\alpha^2 + I_x^2 + I_y^2}.\end{aligned}$$

- Here  $k$  denotes the iteration number,  $v_y^0$  and  $v_x^0$  denote initial velocity estimates which are typically set to zero and  $\bar{u}^k$  and  $\bar{v}^k$  denote neighbourhood averages of  $u^k$  and  $v^k$ . Iterations are stopped if  $k$  reaches a preset value, i.e. 100, or if the norm of the velocity field differences at iterations  $k$  and  $k + 1$  is less than some preset threshold.

# Intensity Differentiation

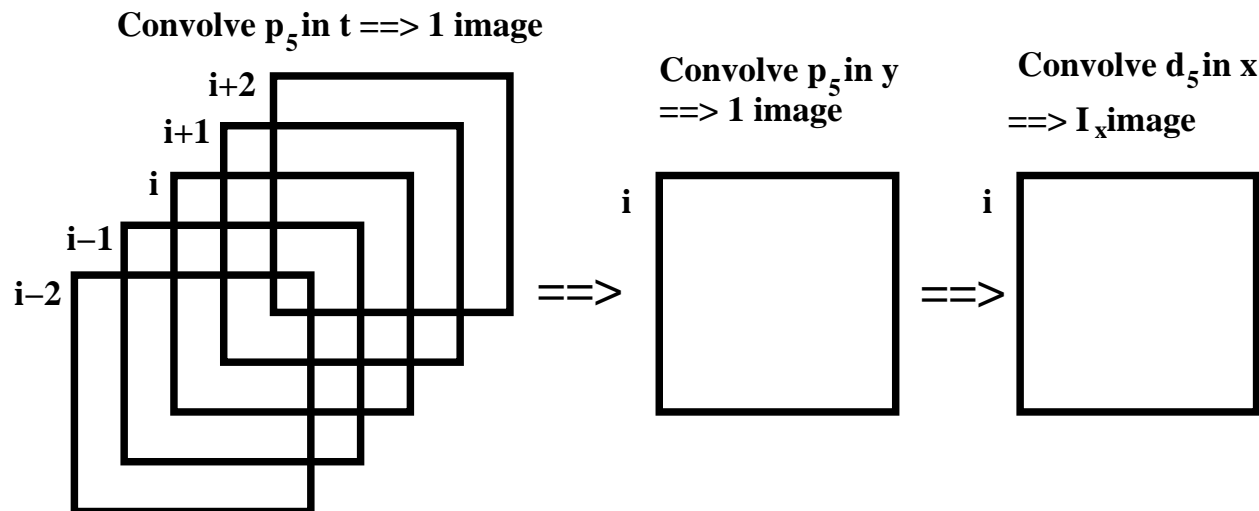
- One way to compute intensity derivatives is via convolution of Simoncelli's Matched/balanced filters [ICIP1994] compute  $I_x$ ,  $I_y$  and  $I_t$ .
- The filter coefficients are:

$n$	$p_5$	$d_5$
0	0.036	-0.108
1	0.249	-0.283
2	0.431	0.0
3	0.249	0.283
4	0.036	0.108

Table 1: Simoncelli's 5-point Matched/Balanced Kernels

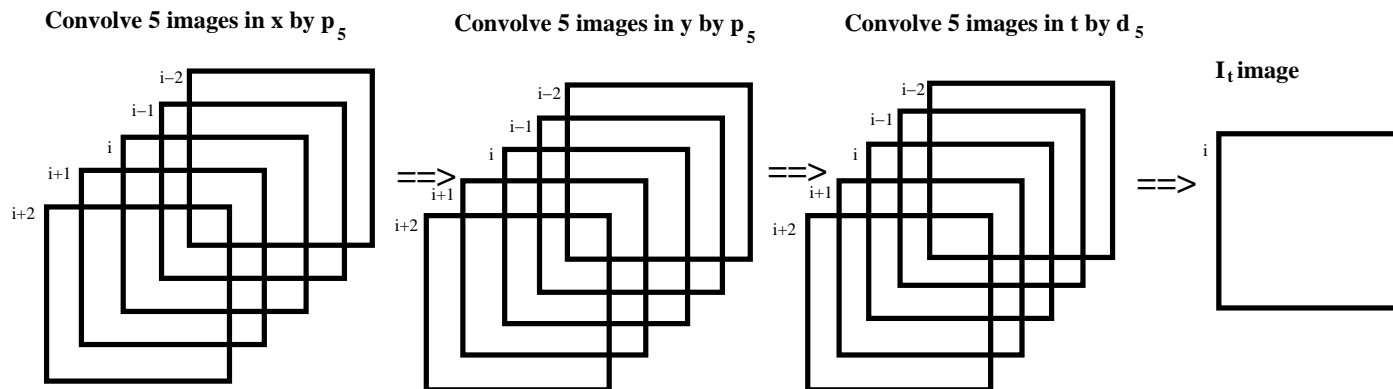
## Convolution Steps for Computing $I_x$

- To compute  $I_x$  in 2D for frame  $i$  is some sequence, we first convolve the smoothing kernel,  $p_5$ , in the  $t$  dimension to reduce the 5 images to 1 image, then convolve the smoothing kernel  $p_5$  on that result in the  $y$  dimension and finally convolve the differentiation kernel,  $d_5$ , on that  $2^{nd}$  result in the  $x$  dimension to obtain  $I_x$ .  $I_y$  is computed in a similar way.



# Convolution Steps for Computing $I_t$

- To compute  $I_t$  in 2D for frame  $i$  in some sequence, we first convolve  $p_5$  in the  $x$  dimension and then on that result in the  $y$  dimension for each of frames  $i - 2, i - 1, i, i + 1$  and  $i + 2$ . This yields 5 smoothed images in  $x$  and  $y$ . We then differentiate these images in the  $t$  dimension using  $d_5$  to get  $I_t$  at frame  $i$ .



## 3D Optical Flow

- Usually, by 3D optical flow we mean 3D volumetric flow: at voxel,  $(x, y, z)$ , what is the 3D velocity  $(U, V, W)$ ? An example of volumetric data: 20 volumes of gated MRI data of 1 beat of a human heart. The 3D motion constraint equations uses intensity derivatives in  $x$ ,  $y$  and  $z$ , as well as  $t$ .
- 3D optical flow on a moving surface (such as a growing leaf) is called **range** flow and also produces 3D velocity  $(U, V, W)$  on all surface points of some scanned object that is moving over time. Now the 3D constraint equation uses  $x$  and  $y$  derivatives of  $Z$ , the depth value (either scanned or computed) at each surface point. Range flow is also often called **scene** flow.

The **3D Intensity Motion Constraint Equation** is a simple extension of the 2D motion constraint equation. Consider a small 3D  $n \times n \times n$  block at  $(x, y, z)$  at time  $t$  moving to  $(x + \delta x, y + \delta y, z + \delta z)$  at time  $t + \delta t$ .



Figure 6: A small  $n \times n \times n$  3D neighbourhood of voxels centered at  $(x, y, z)$  at time  $t$  moving to  $(x + \delta x, y + \delta y, z + \delta z)$  at time  $t + \delta t$ .

## 3D Intensity Motion Constraint Equation and the 3D Aperture Problem

- We assume a 3D voxel  $I(x, y, z)$  at time  $t$  moves with a displacement  $(\delta x, \delta y, \delta z)$  over time  $\delta t$ . Since  $I(x, y, z, t)$  and  $I(x + \delta x, y + \delta y, z + \delta z, t + \delta t)$  are the same we can perform a 1<sup>st</sup> order Taylor series expansion and obtain (as in the 2D case):

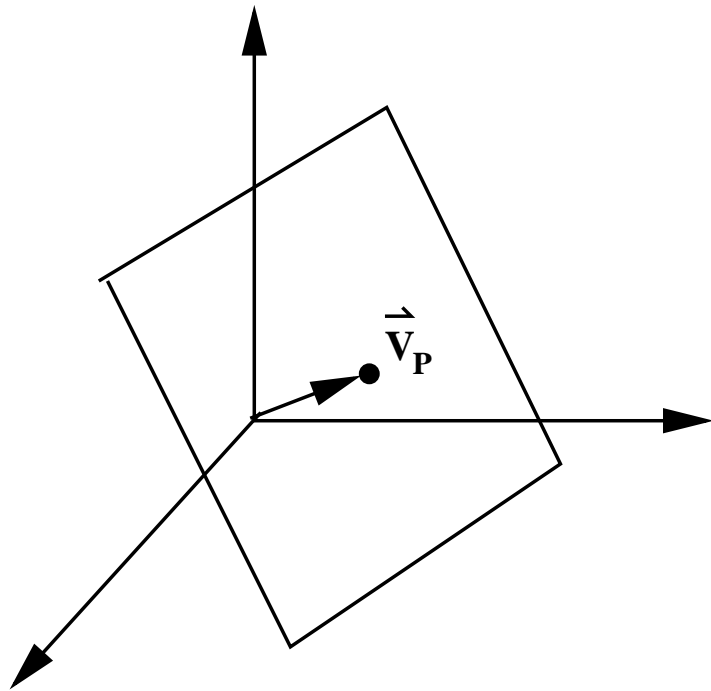
$$I_x U + I_y V + I_z W = \nabla I \cdot \vec{V} = -I_t,$$

$I_x, I_y, I_z$  and  $I_t$  are 3D spatio-temporal intensity derivatives computed via Simoncelli convolution.

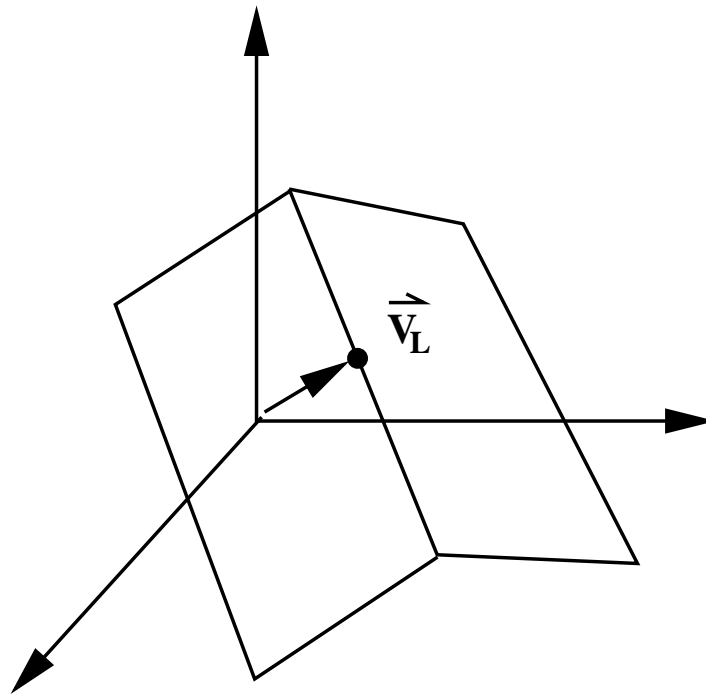
- The 3D velocity is  $\vec{V} = (U, V, W)$
- This equation describes a plane in 3D space. Any point on that plane is possibly the correct 3D velocity. The velocity on the plane that is

closest to the origin is called the **plane normal** velocity (the velocity normal to a local intensity planar structure)

- The **line normal** velocity is the velocity on the line caused by the intersection of 2 planes that is closest to the origin. of course, if three planes intersect at a single point, that point is the full 3D velocity.



3D Plane Normal Velocity



3D Line Normal Velocity

Figure 7: Graphical illustrations of the 3D plane and line normal velocities.

## 3D Lucas and Kanade

- From the 3D the motion constraint equation we have:

$$I_x U + I_y V + I_z W = -I_t,$$

where  $I_x, I_y, I_z$  and  $I_t$  are the 3D intensity derivatives in a  $n \times n \times n$  neighbourhood centered at voxel  $(x, y, z)$  and  $\vec{V} = (U, V, W)$  is that neighbourhood's (assumed) constant 3D velocity.

- Given  $I_x, I_y, I_z$  and  $I_t$  at a single pixel, we can compute  $\vec{V}_n$  as  $\vec{V}_n = \vec{V} \cdot \hat{n}$ , where  $V_n = \frac{-I_t}{\|\nabla I\|_2}$ ,  $\hat{n} = \frac{\nabla I}{\|\nabla I\|_2}$  and  $\nabla I = (I_x, I_y, I_z)$ .
- Given an  $k = n \times n \times n$  neighbourhood with the same velocity  $\vec{V}$  we

can write

$$\underbrace{\begin{bmatrix} n_{x1} & n_{y1} & n_{z1} \\ n_{x2} & n_{y2} & n_{z2} \\ \vdots & \vdots & \vdots \\ n_{xk} & n_{yk} & n_{zk} \end{bmatrix}}_N \underbrace{\begin{bmatrix} U \\ V \\ W \end{bmatrix}}_{\vec{V}} = \underbrace{\begin{bmatrix} V_{n1} \\ V_{n2} \\ \vdots \\ V_{nk} \end{bmatrix}}_B$$

which we can write as  $\underbrace{N}_{k \times 3} \underbrace{\vec{V}}_{3 \times 1} = \underbrace{B}_{k \times 1}$ .

- For  $k \geq 3$  we can solve this system as  $\vec{V} = (N^T N)^{-1} N^T B$ .

## 3D Horn and Schunck

- 3D Horn and Schunck regularization becomes:

$$F = \int_D (I_x U + I_y V + I_z W + I_t)^2 + \alpha^2 (U_x^2 + U_y^2 + U_z^2 + V_x^2 + V_y^2 + V_z^2 + W_x^2 + W_y^2 + W_z^2).$$

- We use the Euler-Lagrange equations:

$$F_U - \frac{d}{dX} F_{U_X} - \frac{d}{dY} F_{U_Y} - \frac{d}{dZ} F_{U_Z} - \frac{d}{dt} F_{U_t} = 0,$$

$$F_V - \frac{d}{dX} F_{V_X} - \frac{d}{dY} F_{V_Y} - \frac{d}{dZ} F_{V_Z} - \frac{d}{dt} F_{V_t} = 0,$$

$$F_W - \frac{d}{dX} F_{W_X} - \frac{d}{dY} F_{W_Y} - \frac{d}{dZ} F_{W_Z} - \frac{d}{dt} F_{W_t} = 0.$$

where:

$$F_U = 2Z_X(Z_X U + Z_Y V + Z_Z W + Z_t)$$

$$F_V = 2Z_Y(Z_X U + Z_Y V + Z_Z W + Z_t)$$

$$F_W = 2Z_Z(Z_X U + Z_Y V + Z_Z W + Z_t)$$

$$F_{U_X} = 2\alpha^2 U_X$$

$$F_{U_Y} = 2\alpha^2 U_Y$$

$$F_{U_Z} = 2\alpha^2 U_Z$$

$$F_{U_t} = 2\alpha^2 U_t$$

$$F_{V_X} = 2\alpha^2 V_X$$

$$F_{V_Y} = 2\alpha^2 V_Y$$

$$F_{V_Z} = 2\alpha^2 V_Z$$

$$F_{V_t} = 2\alpha^2 V_t$$

$$F_{W_X} = 2\alpha^2 W_X$$

$$F_{W_Y} = 2\alpha^2 W_Y$$

$$F_{W_Z} = 2\alpha^2 W_Z$$

$$F_{W_t} = 2\alpha^2 W_t$$

and

$$\begin{aligned}\frac{dF_{U_X}}{dX} &= 2\alpha^2 U_{XX}, & \frac{dF_{V_Z}}{dZ} &= 2\alpha^2 V_{ZZ} \\ \frac{dF_{U_Y}}{dY} &= 2\alpha^2 U_{YY}, & \frac{dF_{V_t}}{dt} &= 2\alpha^2 V_{tt} \\ \frac{dF_{U_Z}}{dZ} &= 2\alpha^2 U_{ZZ}, & \frac{dF_{W_X}}{dX} &= 2\alpha^2 W_{XX} \\ \frac{dF_{U_t}}{dt} &= 2\alpha^2 U_{tt}, & \frac{dF_{W_Y}}{dY} &= 2\alpha^2 W_{YY} \\ \frac{dF_{V_X}}{dX} &= 2\alpha^2 V_{XX}, & \frac{dF_{W_Z}}{dZ} &= 2\alpha^2 W_{ZZ} \\ \frac{dF_{V_Y}}{dY} &= 2\alpha^2 V_{YY}, & \frac{dF_{W_t}}{dt} &= 2\alpha^2 W_{tt}.\end{aligned}$$

- Since  $\nabla^2 U = U_{XX} + U_{YY} + U_{ZZ} + U_{tt}$ ,  
 $\nabla^2 V = V_{XX} + V_{YY} + V_{ZZ} + V_{tt}$  and  
 $\nabla^2 W = W_{XX} + W_{YY} + W_{ZZ} + W_{tt}$  we can rewrite the

Euler-Lagrange equations as:

$$Z_X^2 U + Z_X Z_Y V + Z_X Z_Z W + Z_X Z_t = \alpha^2 \nabla^2 U$$

$$Z_X Z_Y U + Z_Y^2 V + Z_Y Z_Z W + Z_Y Z_t = \alpha^2 \nabla^2 V$$

$$Z_X Z_Z U + Z_Y Z_Z V + Z_Z^2 W + Z_Z Z_t = \alpha^2 \nabla^2 W$$

- Using  $\nabla^2 U \approx \bar{U} - U$ ,  $\nabla^2 V \approx \bar{V} - V$  and  $\nabla^2 W \approx \bar{W} - W$  we can write:

$$(\alpha^2 + Z_X^2)U + Z_X Z_Y V + Z_X Z_Z W = (\alpha^2 \bar{U} + Z_X Z_t)$$

$$Z_X Z_Y U + (\alpha^2 + Z_Y^2)V + Z_Y Z_Z W = (\alpha^2 \bar{V} + Z_Y Z_t)$$

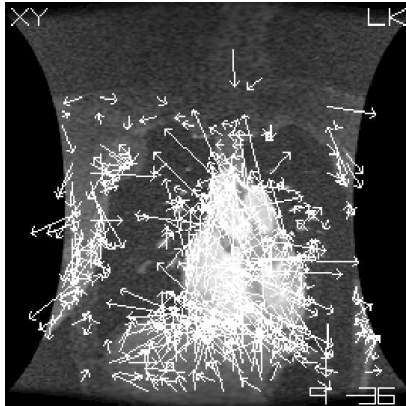
$$Z_X Z_Z U + Z_Y Z_Z V + (\alpha^2 + Z_Z^2)W = (\alpha^2 \bar{W} + Z_Z Z_t).$$

- The Gauss-Seidel iterative equations can be written as:

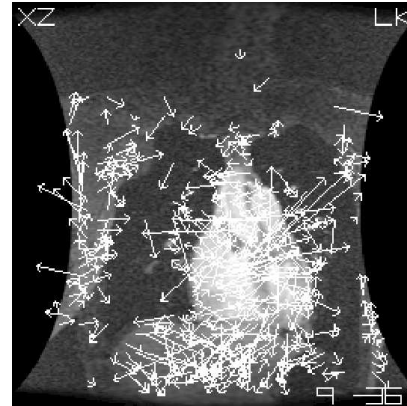
$$\begin{aligned}
 U^{k+1} &= \bar{U}^k - \frac{I_x [I_x \bar{U}^k + I_y \bar{V}^k + I_z \bar{W}^k + I_t]}{(\alpha^2 + I_x^2 + I_y^2 + I_z^2)}, \\
 V^{k+1} &= \bar{V}^k - \frac{I_y [I_x \bar{U}^k + I_y \bar{V}^k + I_z \bar{W}^k + I_t]}{(\alpha^2 + I_x^2 + I_y^2 + I_z^2)} \quad \text{and} \\
 W^{k+1} &= \bar{W}^k - \frac{I_z [I_x \bar{U}^k + I_y \bar{V}^k + I_z \bar{W}^k + I_t]}{(\alpha^2 + I_x^2 + I_y^2 + I_z^2)}.
 \end{aligned}$$

- $\bar{U}^k$ ,  $\bar{V}^k$  and  $\bar{W}^k$  are  $n \times n \times n$  averages of neighbourhoods of velocities at iteration  $k$ .  $\bar{U}^0$ ,  $\bar{V}^0$  and  $\bar{W}^0$  are typically set to 0.0.

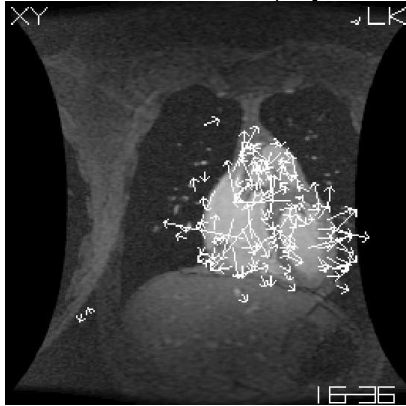
## 3D L&K OF for Gated MRI Cardiac Data



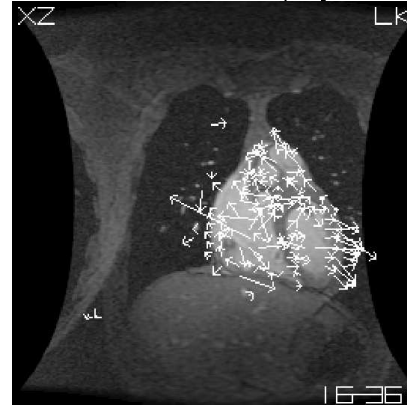
LK-XY-9-36 (5phase)



LK-XZ-9-36 (5phase)



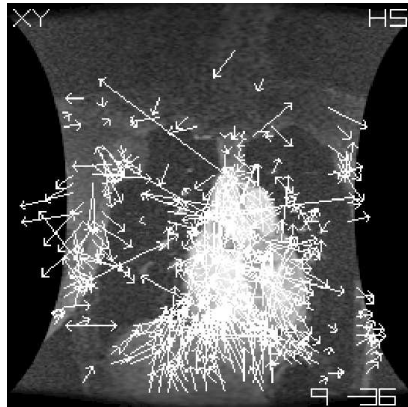
LK-XY-16-36 (10phase)



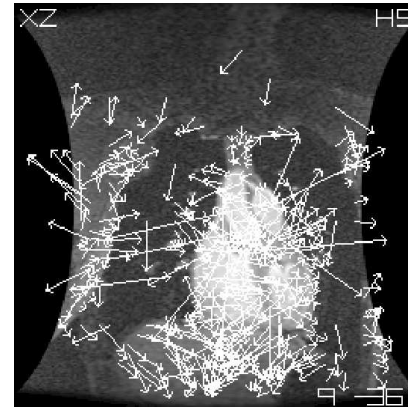
LK-XZ-16-36 (10phase)

**Figure 8:** The Lucas and Kanade  $XY$  and  $XZ$  flow fields superimposed on the  $36^{th}$  slice of the  $9^{th}$  and  $16^{th}$  volumes of the 5phase and 10phase datasets. The eigenvalue threshold  $\lambda_1$  was 1.0.

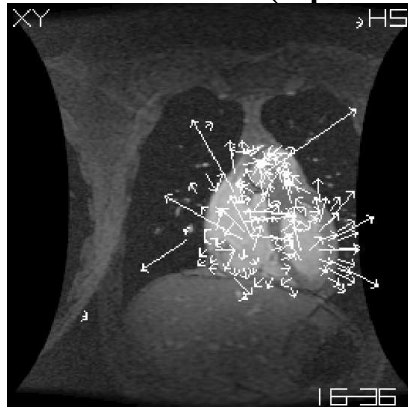
## 3D H&S OF for Gated MRI Cardiac Data



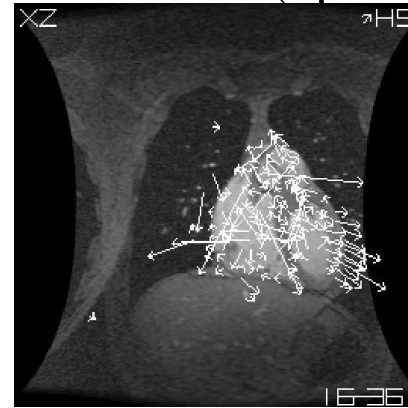
HS-XY-9-36 (5phase)



HS-XZ-9-36 (5phase)



HS-XY-16-36 (10phase)



HS-XZ-16-36 (10phase)

**Figure 9:** The Horn and Schunck  $XY$  and  $XZ$  flow fields superimposed on the  $36^{th}$  slice of the  $9^{th}$  and  $16^{th}$  volumes of the 5phase and 10phase datasets for 100 iterations.  $\alpha = 1.0$ .

## 3D Range Motion Constraint Equation

- We can compute 3D Range Flow (3D Optical Flow on a 3D Surface) from 3D depth data measured by a ShapeGrabber range scanner on rigid and non-rigid surfaces [CVUI2002]. One example is the range flow for a plant leaf (the surface can be deformable) [ECCV2000].
- The 3D Range Constraint Equation is  $Z_x U + Z_y V + W - 1 = 0$ . Here  $Z_x$  and  $Z_y$  are depth derivatives and not intensity derivatives.
- Lucas and Kanade and Horn and Schunck like 3D surface optical flow can be computed [DAGM1999].
- Range flow (from time-varying depth data) can be fused with 2D optical flow (from time-varying intensity data) to produce full 3D velocity on moving flat surfaces, where neither range or optical flow by themselves, can do this [ICIP2000].

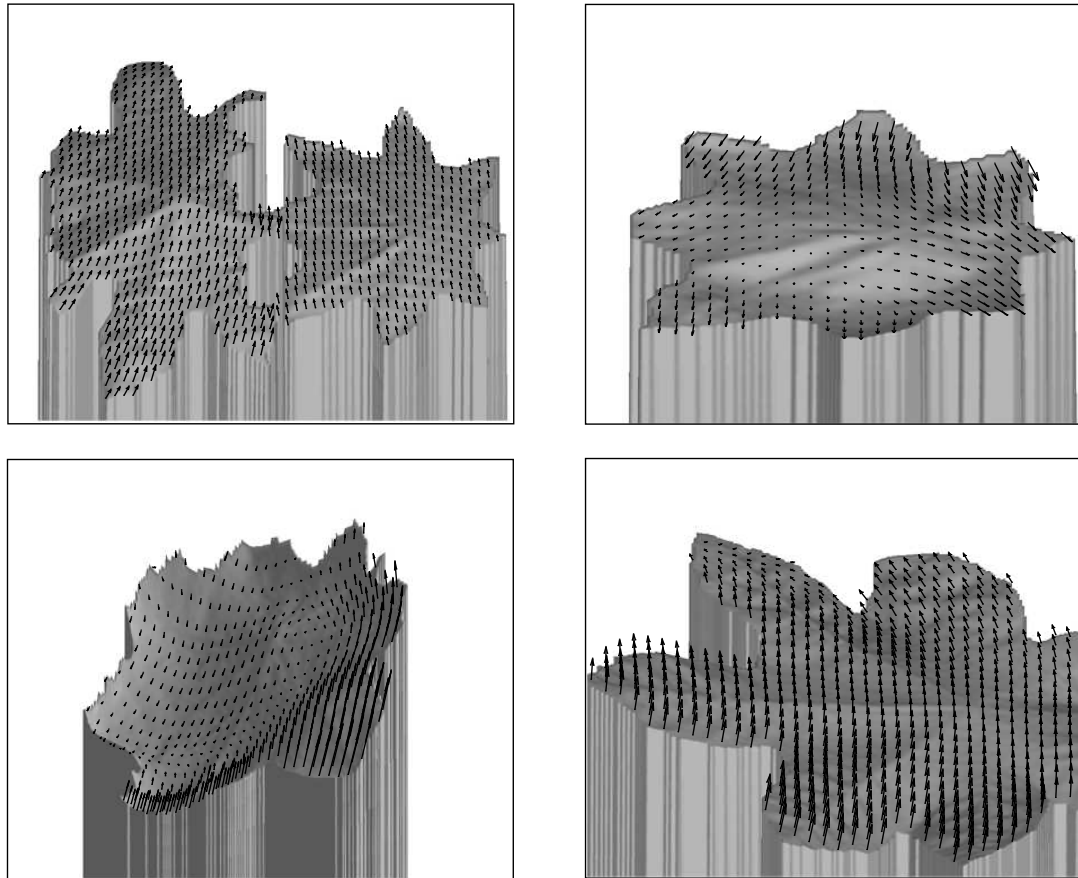


Figure 10: Example 3D range flow for Castor oil plant leaves.

## 3D Scene Flow

- A typical approach: In a stereo image sequence, compute the time-varying depth maps using a stereo algorithm,
- Compute range or scene flow from these time-varying depth maps.
- One point of view: optical flow is not practical on real imaginary but stereo is much better: so use stereo to get depth maps and then compute range flow on these depth maps and their derivative values. One example of such work: “Efficient Dense Scene Flow from Sparse or Dense Stereo Data”, Andreas Wedel, Clemens Rabe, Tobi Vaudrey, Thomas Brox, Uwe Franke and Daniel Cremers, ECCV, October, 2008.