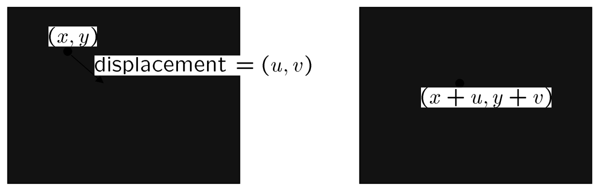
Optical Flow



$H(x, y)$ $\qquad$ $I(x, y)$

- Small motion: (u and v are less than 1 pixel)
  - *H(x,y) = I(x+u,y+v)*
- Brute force not possible
    - suppose we take the Taylor series expansion of I:

$$I(x+u, y+v) = I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$
$$\approx I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

(Seitz)

Optical flow equation

- Combining these two equations

shorthand: $I_x = \frac{\partial I}{\partial x}$

$$\begin{aligned} 0 &= I(x+u, y+v) - H(x, y) \\ &\approx I(x, y) + I_x u + I_y v - H(x, y) \\ &\approx (I(x, y) - H(x, y)) + I_x u + I_y v \\ &\approx I_t + I_x u + I_y v \\ &\approx I_t + \nabla I \cdot [u \;\; v] \end{aligned}$$

- In the limit as u and v go to zero, this becomes exact

$$0 = I_t + \nabla I \cdot [\tfrac{\partial x}{\partial t} \;\; \tfrac{\partial y}{\partial t}]$$

(Seitz)

Optical flow equation

$$0 = I_t + \nabla I \cdot [u\ v]$$

- Q: how many unknowns and equations per pixel?
- Intuitively, what does this constraint mean?
  - The component of the flow in the gradient direction is determined
  - The component of the flow parallel to an edge is unknown

(Seitz)

This explains the Barber Pole illusion
http://www.sandlotscience.com/Ambiguous/barberpole.htm

Let's look at an example of this. Suppose we have an image in which H(x,y) = y. That is, the image will look like:

11111111111111

22222222222222

33333333333333

And suppose there is optical flow of (1,1). The new image will look like:

-1111111111111

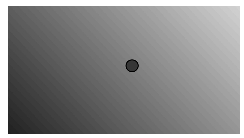
-2222222222222

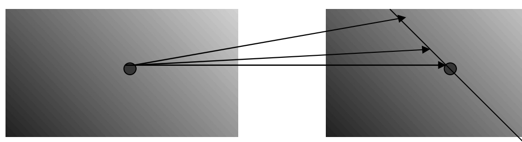
I(3,3) = 2. H(3,3) = 3. So I_t(3,3) = -1. GRAD I(3,3) = (0,1). So our constraint equation will be: 0 = -1 + <(0,1), (u,v)>, which is 1 = v. We recover the v component of the optical flow, but not the u component. This is the aperture problem.

First Order Approximation

When we assume: $I(x+u, y+v) \approx I(x, y) + \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v$

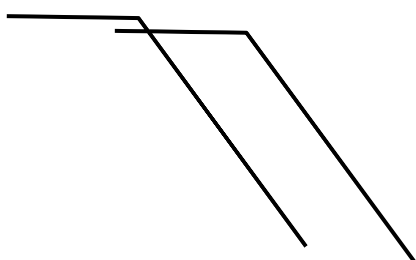
We assume an image locally is:





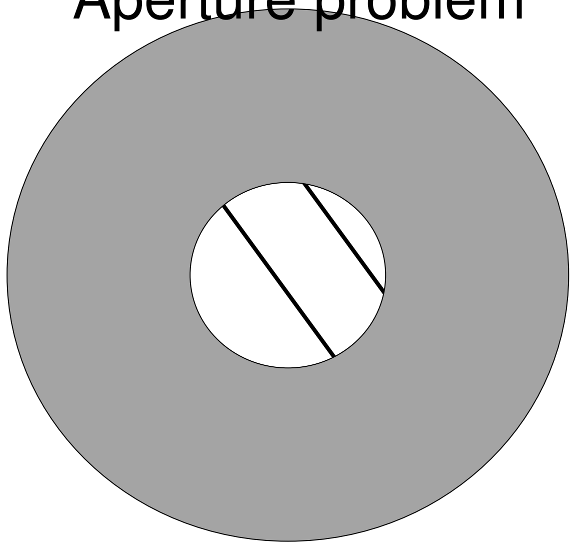
(Seitz)

Aperture problem



(Seitz)

Aperture problem



(Seitz)

Solving the aperture problem

- How to get more equations for a pixel?
  - Basic idea: impose additional constraints
    - most common is to assume that the flow field is smooth locally
    - one method: pretend the pixel's neighbors have the same (u,v)
      - If we use a 5x5 window, that gives us 25 equations per pixel! $0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u \ v]$

$$\underbrace{\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix}}_{\substack{A \\ 25\times 2}} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\substack{d \\ 2\times 1}} = - \underbrace{\begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}}_{\substack{b \\ 25\times 1}}$$

(Seitz)

Lukas-Kanade flow

$$\underset{25\times 2}{A}\ \underset{2\times 1}{d} = \underset{25\times 1}{b} \longrightarrow \text{minimize } ||Ad - b||^2$$

- We have more equations than unknowns: solve least squares problem. This is given by:

$$\overset{2\times 2}{(A^T A)}\ \overset{2\times 1}{d} = \overset{2\times 1}{A^T b}$$

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

– Summations over all pixels in the KxK window

– Does $A^T A$ look familiar?

(Seitz)

Let's look at an example of this. Suppose we have an image with a corner.

1111111111 -----------------

1222222222 And this translates down and to the right: -1111111111

1233333333 -1222222222

1234444444 -1233333333

Let's compute I_t for the whole second image:

---------- Ix = ---------- Iy = -------------

0-1-1-1-1-1 --00000 --------------

-1-1-1-1-1-1 --.50000 -0-.5-1-1-1-1-1-1

-1-1-1-1-1-1- --1.5000 -00-.5-1-1-1-1-1

Then the equations we get have the form:

(.5,-.5)*(u,v) = 1, (1,0)*(u,v) = 1, (0,-1)(u,v) = 1.

Together, these lead to a solution that u = 1, v = -1.

Conditions for solvability

– Optimal (u, v) satisfies Lucas-Kanade equation

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

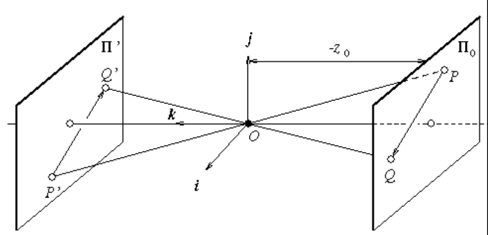
When is This Solvable?

- $\mathbf{A^T A}$ should be invertible
- $\mathbf{A^T A}$ should not be too small due to noise
  - eigenvalues λ_1 and λ_2 of $\mathbf{A^T A}$ should not be too small
- $\mathbf{A^T A}$ should be well-conditioned
  - λ_1 / λ_2 should not be too large (λ_1 = larger eigenvalue)

(Seitz)

Weak perspective (scaled orthographic projection)

- Issue
  - perspective effects, but not over the scale of individual objects
  - collect points into a group at about the same depth, then divide each point by the depth of its group



(Forsyth & Ponce)

Perspective -> Scaled Orthographic

- Recall: $(x_i, y_i, z_i) \rightarrow (x_i/z_i,\ y_i/z_i)$
- Let $Z = (z_1 + z_2 + \ldots + z_n)/n$
- Then, (x_i, y_i, z_i) approx-> $(x_i/Z,\ y_i/Z)$

The Equation of Weak Perspective

$$(x, y, z) \rightarrow s(x, y) \qquad \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} s & 0 & 0 & 0 \\ 0 & s & 0 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

- s is constant for all points.
- Parallel lines no longer converge, they remain parallel.

Pros and Cons of These Models

- Weak perspective much simpler math.
  - Accurate when object is small and distant.
  - Most useful for recognition.
- Pinhole perspective much more accurate for scenes.
  - Used in structure from motion.
- When accuracy really matters, must model real cameras.

First: Represent motion

- We'll talk about a fixed camera, and moving object.
- Key point:

Points

$$P = \begin{pmatrix} x_1 & x_2 & & & & x_n \\ y_1 & y_2 & . & . & . & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

Some matrix

$$S = \begin{pmatrix} s_{1,1} & s_{1,2} & s_{1,3} & t_x \\ s_{2,1} & s_{2,2} & s_{2,3} & t_y \end{pmatrix}$$

The image

$$I = \begin{pmatrix} u_1 & u_2 & . & . & . & u_n \\ v_1 & v_2 & & & & v_n \end{pmatrix}$$

Then: $I = SP$

Structure-from-Motion

- S encodes:
  - Projection: only two lines
  - Scaling, since S can have a scale factor.
  - Translation, by tx/s and ty/s.
  - Rotation:

$$I = SP$$

Rotation

$$\begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1} & r_{2,2} & r_{2,3} \\ r_{3,1} & r_{3,2} & r_{3,3} \end{pmatrix} P$$

Represents a 3D rotation of the points in P.

First, look at 2D rotation (easier)

Matrix R acts on points by rotating them.

$$R = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$$

$$\begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \end{pmatrix}$$

- Also, RR^T = Identity. R^T is also a rotation matrix, in the opposite direction to R.

Simple 3D Rotation

$$\begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \end{pmatrix}$$

Rotation about z axis.

Rotates x,y coordinates. Leaves z coordinates fixed.

Full 3D Rotation

$$R = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{pmatrix}$$

• Any rotation can be expressed as combination of three rotations about three axes.

$$RR^T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

• Rows (and columns) of *R* are orthonormal vectors.

• R has determinant 1 (not -1).

Putting it Together

Scale

3D Translation

$$s\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \end{pmatrix}\begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & 0 \\ r_{2,1} & r_{2,2} & r_{2,3} & 0 \\ r_{3,1} & r_{3,2} & r_{3,3} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}P$$

Projection

3D Rotation

$$\equiv \begin{pmatrix} s_{1,1} & s_{1,2} & s_{1,3} & st_x \\ s_{2,1} & s_{2,2} & s_{2,3} & st_y \end{pmatrix}P$$

where

$$(s_{1,1}, s_{1,2}, s_{1,3}) \bullet (s_{2,1}, s_{2,2}, s_{2,3}) = 0$$

$$\|(s_{1,1}, s_{1,2}, s_{1,3})\| = \|(s_{2,1}, s_{2,2}, s_{2,3})\|$$

We can just write st_x as t_x and st_y as t_y.

Affine Structure from Motion

$$\begin{pmatrix} s_{1,1} & s_{1,2} & s_{1,3} & t_x \\ s_{2,1} & s_{2,2} & s_{2,3} & t_y \end{pmatrix} P$$

~~where~~

$$\cancel{(s_{1,1}, s_{1,2}, s_{1,3}) \bullet (s_{2,1}, s_{2,2}, s_{2,3}) = 0}$$

$$\cancel{\left\|(s_{1,1}, s_{1,2}, s_{1,3})\right\| = \left\|(s_{2,1}, s_{2,2}, s_{2,3})\right\|}$$

Affine Structure-from-Motion

$$\begin{pmatrix} u_1^1 & u_2^1 & . & . & . & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \\ . & . & & & & . \\ . & . & & & & . \\ . & . & & & & . \\ u_1^m & u_2^m & & & & u_n^m \\ v_1^m & v_2^m & . & . & . & v_n^m \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \\ & . & & \\ & . & & \\ & . & & \\ s_{1,1}^m & s_{1,2}^m & s_{1,3}^m & t_x^m \\ s_{2,1}^m & s_{2,2}^m & s_{2,3}^m & t_y^m \end{pmatrix} \begin{pmatrix} x_1 & x_2 & . & . & . & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

I S P

First Step: Solve for Translation (1)

- This is trivial, because we can pick a simple origin.
  - World origin is arbitrary.
  - Example: We can assume first point is at origin.
    - Rotation then doesn't effect that point.
    - All its motion is translation.
  - Better to pick center of mass as origin.
    - Average of all points.
    - This also averages all noise.

First Step: Solve for Translation (2)

$$\text{WLOG}: \sum_{i=1}^{n} \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\bar{u}^i = \sum_{k=1}^{n} u_k^i \Big/ n$$

$$\bar{v}^i = \sum_{k=1}^{n} v_k^i \Big/ n$$

$$\tilde{I} = \begin{pmatrix} u_1^1 - \bar{u}^1 & u_2^1 - \bar{u}^1 & . \quad . \quad . & u_n^1 - \bar{u}^1 \\ v_1^1 - \bar{v}^1 & v_2^1 - \bar{v}^1 & & v_n^1 - \bar{v}^1 \\ u_1^2 - \bar{u}^2 & u_2^2 - \bar{u}^2 & & u_n^2 - \bar{u}^2 \\ v_1^2 - \bar{v}^2 & v_2^2 - \bar{v}^2 & & v_n^2 - \bar{v}^2 \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ u_1^m - \bar{u}^m & u_2^m - \bar{u}^m & & u_n^m - \bar{u}^m \\ v_1^m - \bar{v}^m & v_2^m - \bar{v}^m & . \quad . \quad . & v_n^m - \bar{v}^m \end{pmatrix}$$

First Step: Solve for Translation (3)

$$\begin{pmatrix} \tilde{u}_1^1 & \tilde{u}_2^1 & \cdots & \tilde{u}_n^1 \\ \tilde{v}_1^1 & \tilde{v}_2^1 & & \tilde{v}_n^1 \\ \tilde{u}_1^2 & \tilde{u}_2^2 & & \tilde{u}_n^2 \\ \tilde{v}_1^2 & \tilde{v}_2^2 & & \tilde{v}_n^2 \\ \vdots & \vdots & & \vdots \\ \tilde{u}_1^m & \tilde{u}_2^m & & \tilde{u}_n^m \\ \tilde{v}_1^m & \tilde{v}_2^m & \cdots & \tilde{v}_n^m \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 \\ & \vdots & \\ s_{1,1}^m & s_{1,2}^m & s_{1,3}^m \\ s_{2,1}^m & s_{2,2}^m & s_{2,3}^m \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & & y_n \\ z_1 & z_2 & & z_n \end{pmatrix}$$

As if by magic, there’s no translation.

Rank Theorem

$$\underset{\tilde{I}}{\begin{pmatrix} \tilde{u}_1^1 & \tilde{u}_2^1 & \cdots & \tilde{u}_n^1 \\ \tilde{v}_1^1 & \tilde{v}_2^1 & & \tilde{v}_n^1 \\ \tilde{u}_1^2 & \tilde{u}_2^2 & & \tilde{u}_n^2 \\ \tilde{v}_1^2 & \tilde{v}_2^2 & & \tilde{v}_n^2 \\ \vdots & \vdots & & \vdots \\ \tilde{u}_1^m & \tilde{u}_2^m & & \tilde{u}_n^m \\ \tilde{v}_1^m & \tilde{v}_2^m & \cdots & \tilde{v}_n^m \end{pmatrix}} = \underset{S}{\begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 \\ & \vdots & \\ s_{1,1}^m & s_{1,2}^m & s_{1,3}^m \\ s_{2,1}^m & s_{2,2}^m & s_{2,3}^m \end{pmatrix}} \underset{P}{\begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & & y_n \\ z_1 & z_2 & & z_n \end{pmatrix}}$$

$\tilde{I}$ has rank 3.

This means there are 3 vectors such that every row of $\tilde{I}$ is a linear combination of these vectors. These vectors are the rows of P.

Solve for S

- SVD is made to do this.

$\tilde{I} = UDV$

D is diagonal with non-increasing values.

U and V have orthonormal rows.

Ignoring values that get set to 0, we have U(:,1:3) for S, and

D(1:3,1:3)*V(1:3,:) for P.

Linear Ambiguity

$\tilde{I}$ = U(:,1:3) * D(1:3,1:3) * V(1:3,:)

$\tilde{I}$ = (U(:,1:3) * A) * (inv(A) *D(1:3,1:3) * V(1:3,:))

Noise

- $\tilde{I}$ has full rank.
- Best solution is to estimate I that's as near to $\tilde{I}$ as possible, with estimate of I having rank 3.
- Our current method does this.

Weak Perspective Motion

$$
\begin{pmatrix}
\tilde{u}_1^1 & \tilde{u}_2^1 & . & . & . & \tilde{u}_n^1 \\
\tilde{v}_1^1 & \tilde{v}_2^1 & & & & \tilde{v}_n^1 \\
\tilde{u}_1^2 & \tilde{u}_2^2 & & & & \tilde{u}_n^2 \\
\tilde{v}_1^2 & \tilde{v}_2^2 & & & & \tilde{v}_n^2 \\
. & . & & & & . \\
. & . & & & & . \\
. & . & & & & . \\
\tilde{u}_1^m & \tilde{u}_2^m & & & & \tilde{u}_n^m \\
\tilde{v}_1^m & \tilde{v}_2^m & . & . & . & \tilde{v}_n^m
\end{pmatrix}
=
\begin{pmatrix}
s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 \\
s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 \\
s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 \\
s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 \\
 & . & \\
 & . & \\
 & . & \\
s_{1,1}^m & s_{1,2}^m & s_{1,3}^m \\
s_{2,1}^m & s_{2,2}^m & s_{2,3}^m
\end{pmatrix}
\begin{pmatrix}
x_1 & x_2 & . & . & . & x_n \\
y_1 & y_2 & & & & y_n \\
z_1 & z_2 & & & & z_n
\end{pmatrix}
$$

$\tilde{I}$ S P

$\tilde{I}$ =(U(:,1:3)*A)*(inv(A) *D(1:3,1:3)*V(1:3,:))

Row 2k and 2k+1 of S should be orthogonal. All rows should be unit vectors.

(Push all scale into P).

Choose A so (U(:,1:3) * A) satisfies these conditions.

Multi-object Motion

$$S_k = \text{Motion of Object } k$$

$$P_k = \text{Points of Object } k$$

$$I = [S_1 \mid S_2 \mid \ldots \mid S_n] \begin{bmatrix} P_1 & 0 & . & 0 \\ 0 & P_2 & . & 0 \\ . & . & . & . \\ 0 & 0 & . & P_n \end{bmatrix}$$

Given enough images, *I* will have rank 4n.

Multi-body Factorization

- Main insight. Consider columns of I, corresponding to individual points.
  - If two columns come from different objects, they are unrelated. Their inner product is likely to be small.
  - If five columns come from the same object, in general the fifth will be a linear combination of the other four.

Factorization

We rewrite the images, I, in a different coordinate system.

$I_k =$ column k of I

Rotate space so that the first coordinate is in the direction of I_1

$$I^1 = \begin{pmatrix} 1 & I^1_{1,2} & I^1_{1,3} & . \\ 0 & I^1_{2,2} & . & . \\ 0 & . & . & . \\ 0 & . & . & . \end{pmatrix}$$

The superscript denotes the fact that we have performed one rotation. After this rotation, the first column has coordinates $(1,0,0,...)$ and the other columns have arbitrary coordinates.

Next we rotate so the second coordinate is in the direction of that component of I_2 that is orthogonal to I_1.

$$I^2 = \begin{pmatrix} 1 & I^2_{1,2} & I^2_{1,3} & . \\ 0 & I^1_{2,2} & . & . \\ 0 & 0 & . & . \\ 0 & 0 & . & . \end{pmatrix}$$

As we continue this process, we reach a column that belongs to the same object as four previous columns. This column cannot provide a new axis of the coordinate system. Instead, its coordinates will be non-zero in the rows that are non-zero for the four previous columns from this object, and will be zero (or small) in other rows. This allows us to readily detect the separately moving objects in the scene.