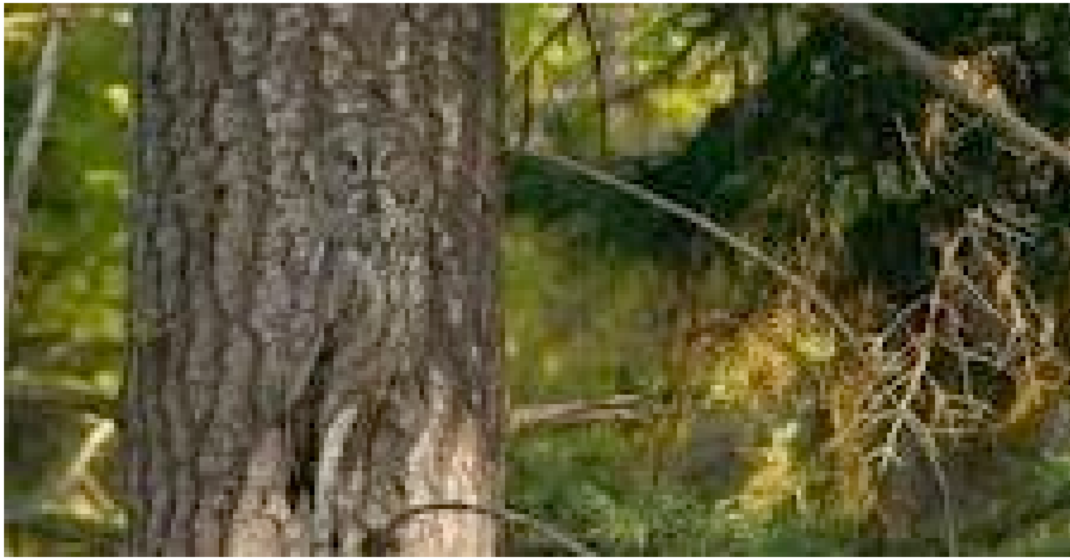
Motion Extraction

Motion is a basic cue

Motion can be the only cue for segmentation

Biologically favoured because of camouflage



Computer Vision

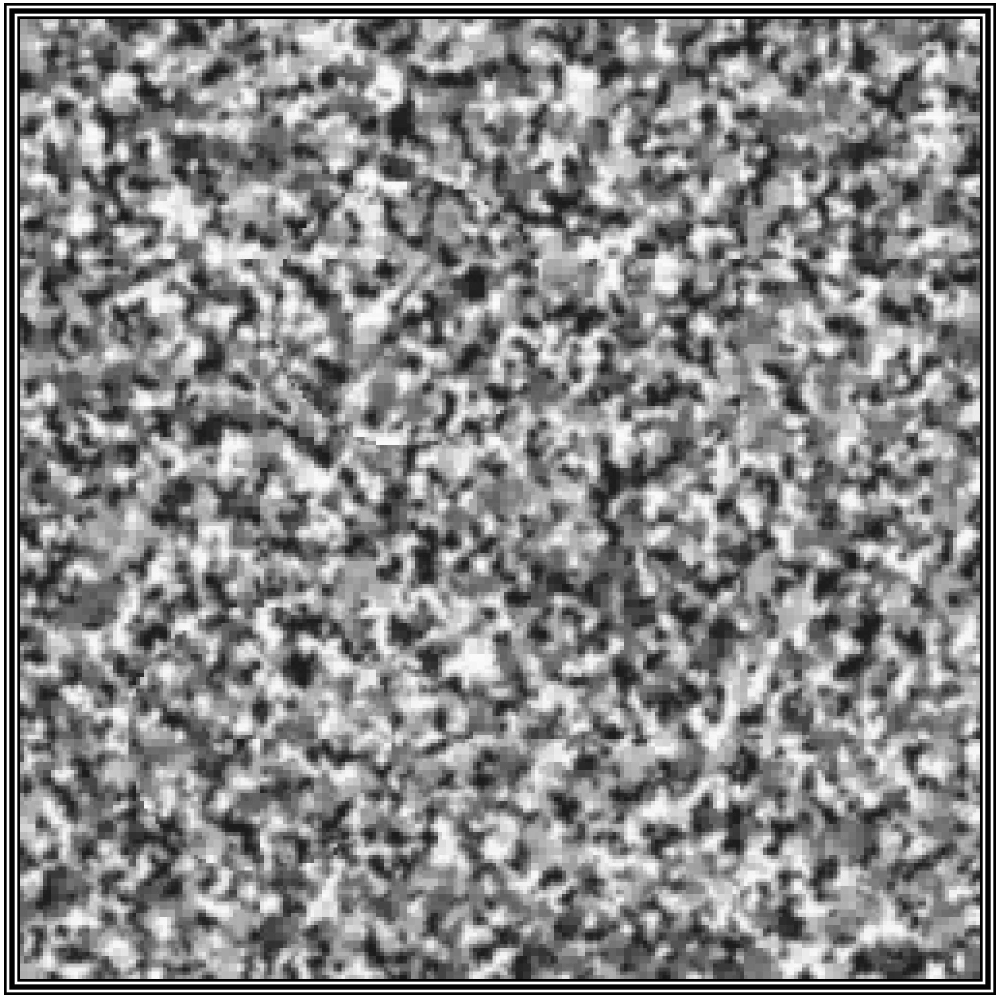
Motion is a basic cue

… which set in motion a constant, evolutionary race

Computer Vision

Motion is a basic cue

Motion can be the only cue for segmentation



Motion is a basic cue

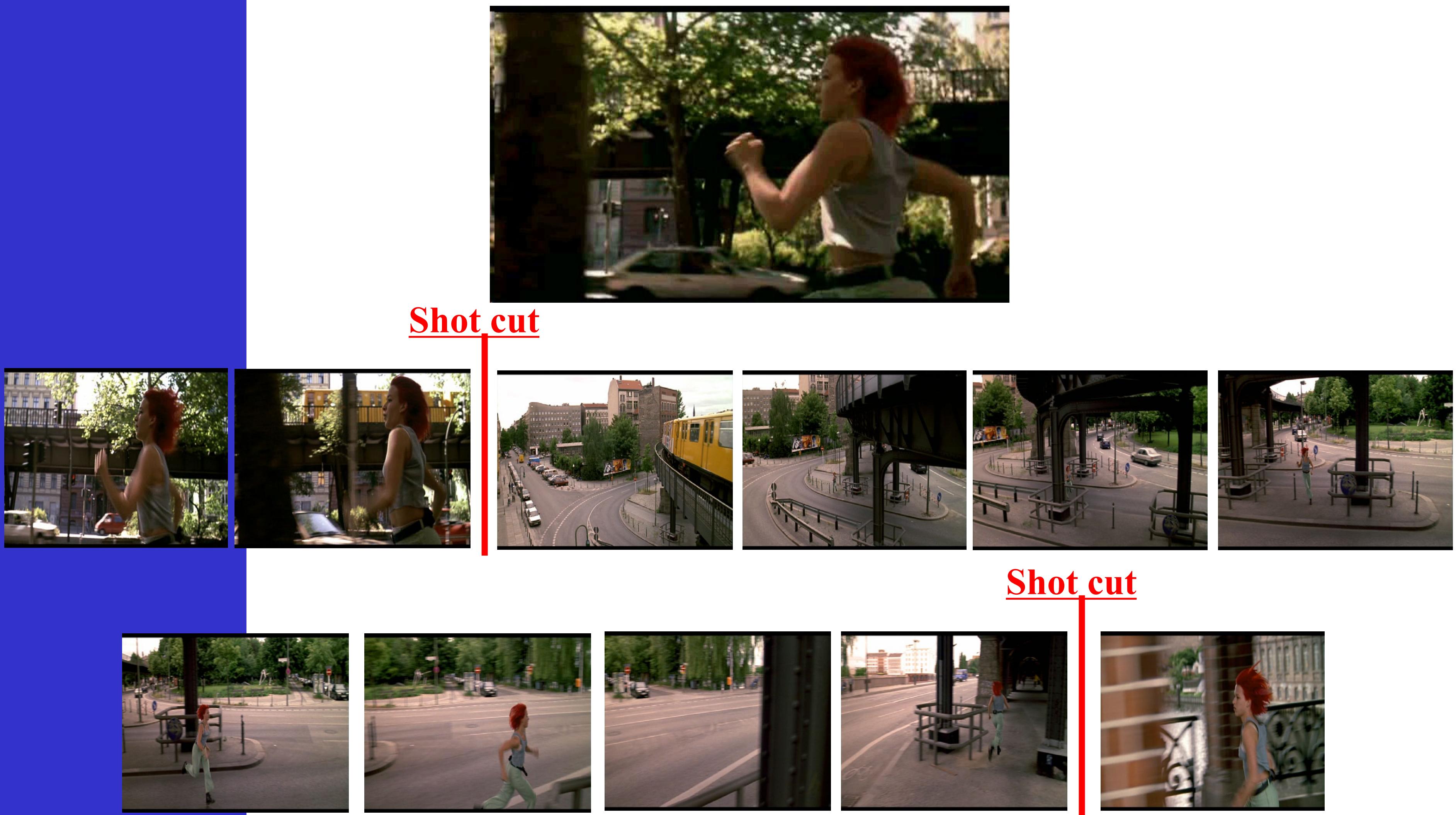
Even impoverished motion data can elicit a strong percept

http://www.biomotionlab.ca/Demos/BMLwalker.html

Some applications of motion extraction

- Change / shot cut detection
- Surveillance / traffic monitoring
- Autonomous driving
- Analyzing game dynamics in sports
- Motion capture / gesture analysis (HCI)
- Image stabilisation
- Motion compensation (e.g. medical robotics)
- Feature tracking for 3D reconstruction
- **Etc. !**

Shot cut detection & Keyframes



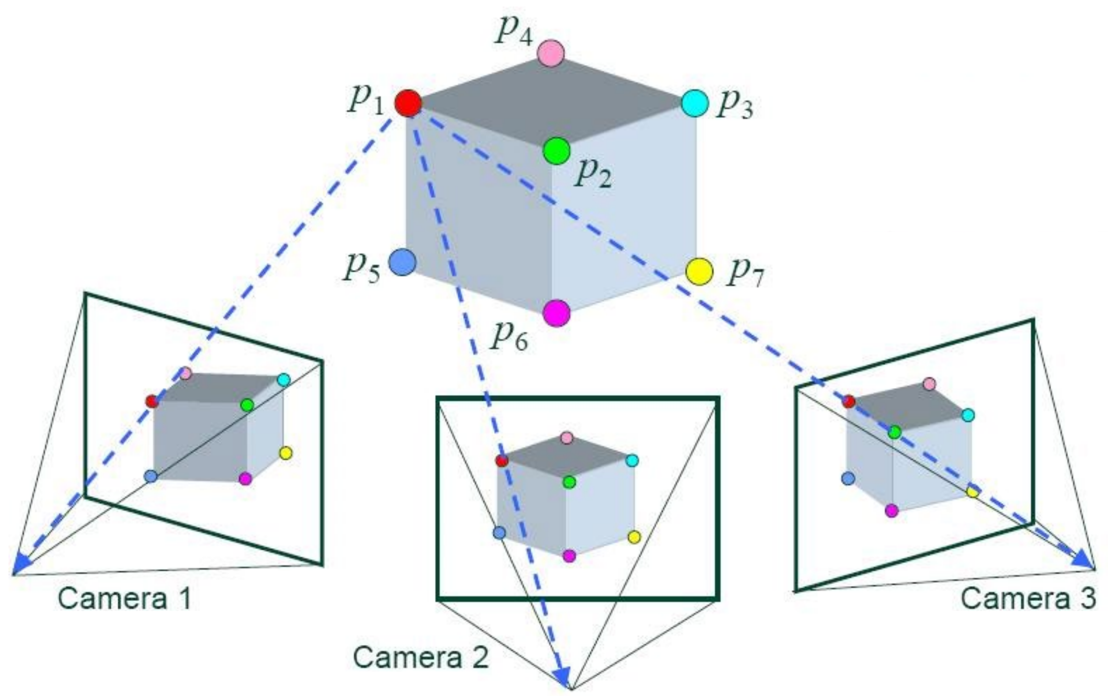
Computer Vision

Human-Machine Interfacing

KINECT for XBOX 360

3D: Structure-from-Motion

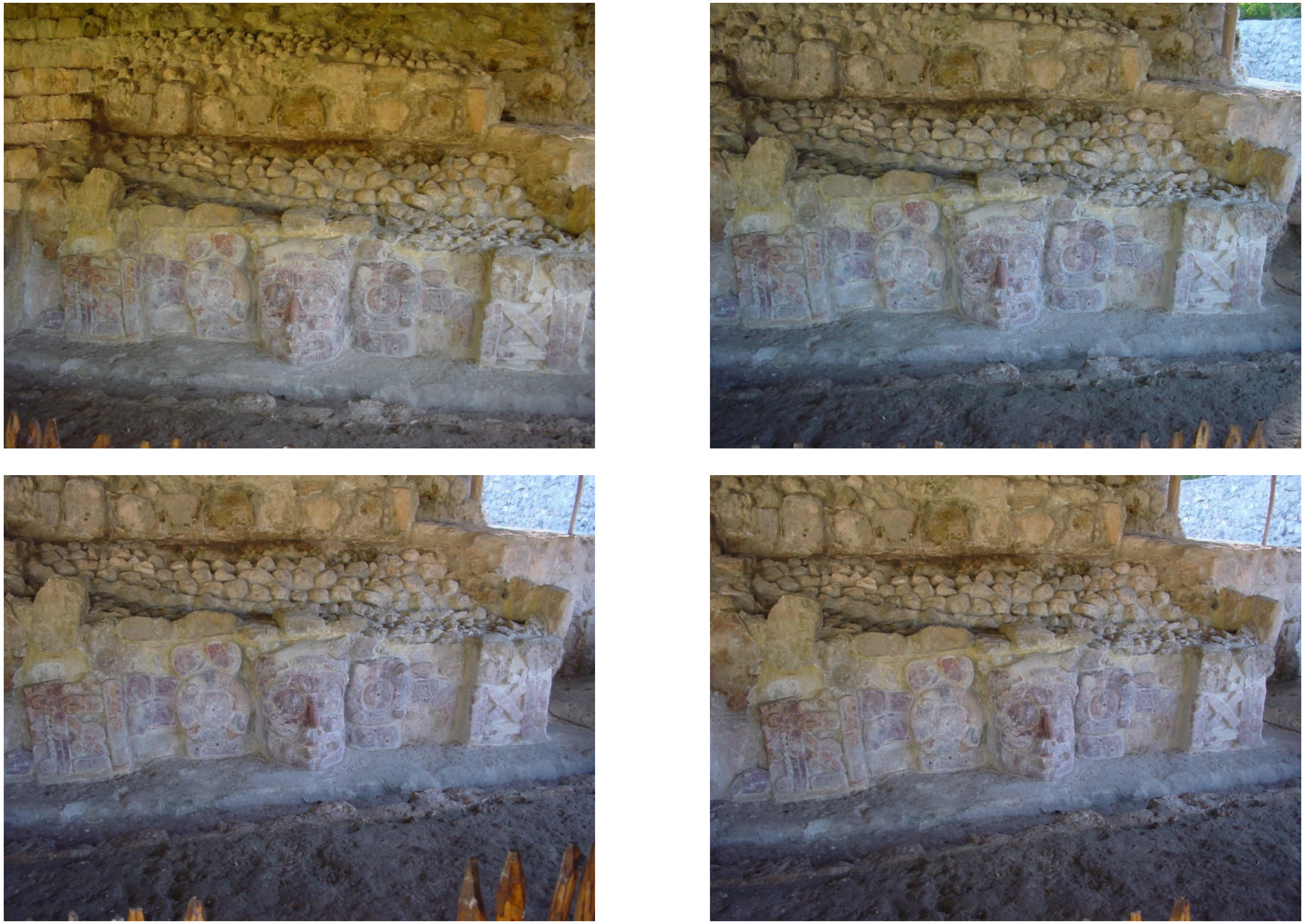
Tracked Points gives correspondences



Computer Vision

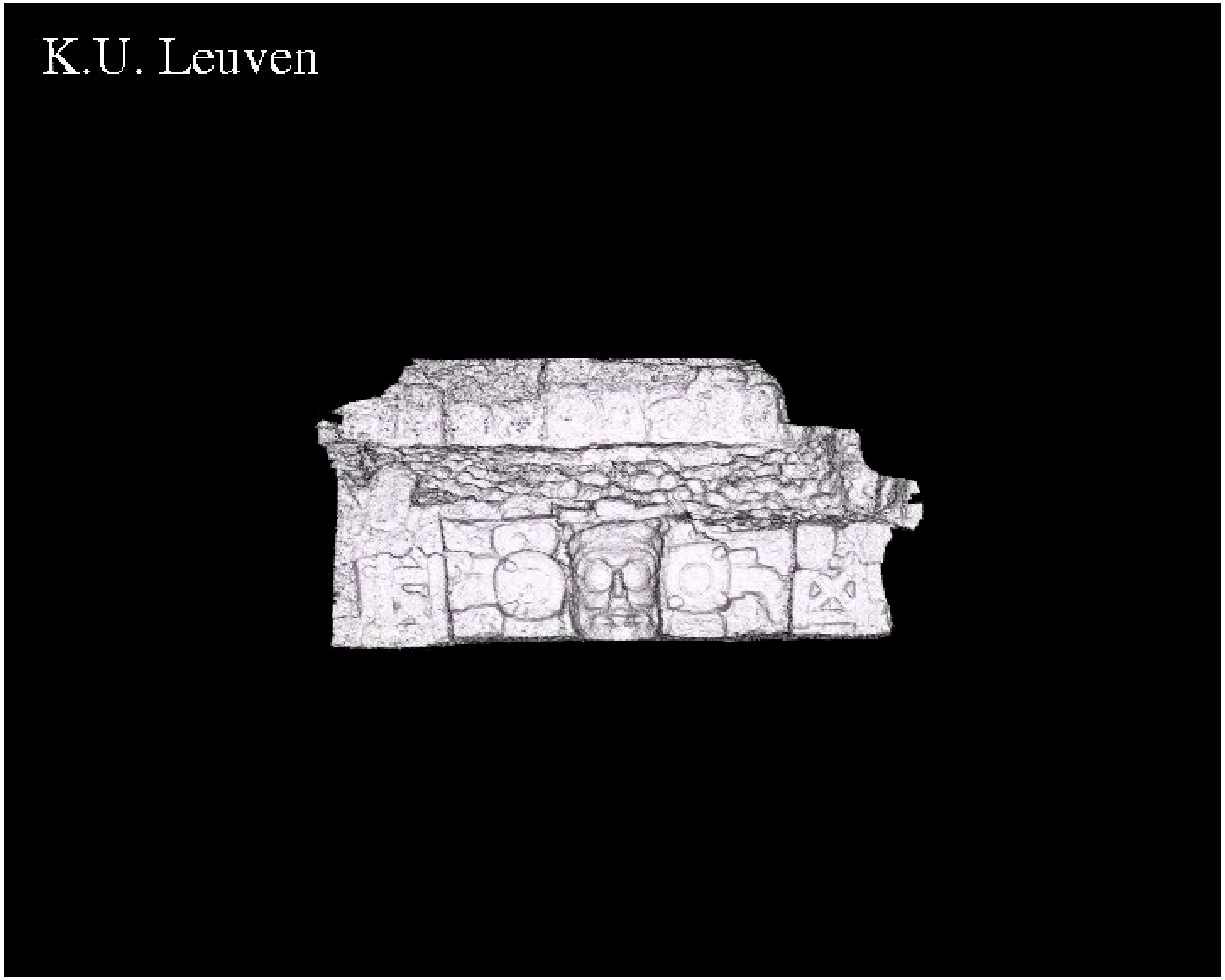
3D: Structure-from-Motion

Temple of the Masks, Edzna, Mexico



Computer Vision

www.arc3d.b



in this lecture...

Several techniques, but…
this lecture is restricted to the

1. detection of the "optical flow"

2. tracking with the "Condensation filter"

optical flow

Definition of optical flow

OPTICAL FLOW = apparent motion of brightness patterns

Ideally, the optical flow is the projection of the three-dimensional motion vectors on the image

Such 2D motion vector is sought at every pixel of the image (note: a motion vector here is a 2D translation vector)

Caution required !

Two examples where following brightness patterns is misleading:

1. Untextured, rotating sphere

$$\Downarrow$$

$$O.F. = 0$$

2. No motion, but changing lighting

$$\Downarrow$$

$$O.F. \neq 0$$

Caution required !

Qualitative formulation

Suppose a *point of the scene* projects to a certain pixel of the current video frame. Our task is to figure out to which pixel in the next frame it moves…

That question needs answering *for all pixels* of the current image.

In order to find these corresponding pixels, we need to come up with a reasonable assumption on how we can detect them among the many.

We assume these corresponding pixels have the *same intensities* as the pixels the scene points came from in the previous frame.

That will only hold approximately…

Mathematical formulation

Our mathematical representation of a video:

$I\ (x,y,t)$ = brightness at (x,y) at time t

Optical flow constraint equation :

$$\frac{dI}{dt} = \frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$$

This equation states that if one were to track the image projections of a scene point through the video, it would not change its intensity. This tends to be true over short lapses of time.

Mathematical formulation

Our mathematical representation of a video:

$I\ (x,y,t)$ = brightness at (x,y) at time t

Optical flow constraint equation :

$$\boxed{\frac{dI}{dt}} = \frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \boxed{\frac{\partial I}{\partial t}} = 0$$

Note the different types of time derivatives !

Mathematical formulation

Our mathematical representation of a video:

$I\ (x,y,t)$ = brightness at (x,y) at time t

Optical flow constraint equation :

$$\frac{dI}{dt} = \frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$$

Change of intensity when following a physical point through the images

Change of intensity when looking at the same pixel (x,y) through the images

Mathematical formulation

We will use as shorthand notation for

$$\frac{dI}{dt} = \frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$$

$$u = \frac{dx}{dt}, \qquad v = \frac{dy}{dt}$$

$$I_x = \frac{\partial I}{\partial x}, \quad I_y = \frac{\partial I}{\partial y}, \quad I_t = \frac{\partial I}{\partial t}$$

$$I_x u + I_y v + I_t = 0$$

1 equation per pixel

Computer Vision

The aperture problem

$$u = \frac{dx}{dt}, \qquad v = \frac{dy}{dt}$$

$$I_x = \frac{\partial I}{\partial x}, \quad I_y = \frac{\partial I}{\partial y}, \quad I_t = \frac{\partial I}{\partial t}$$

$$I_x u + I_y v + I_t = 0$$

Note that we can measure the 3 derivatives of I, but that u and v are unknown

1 equation in 2 unknowns… the `aperture problem'

Computer Vision

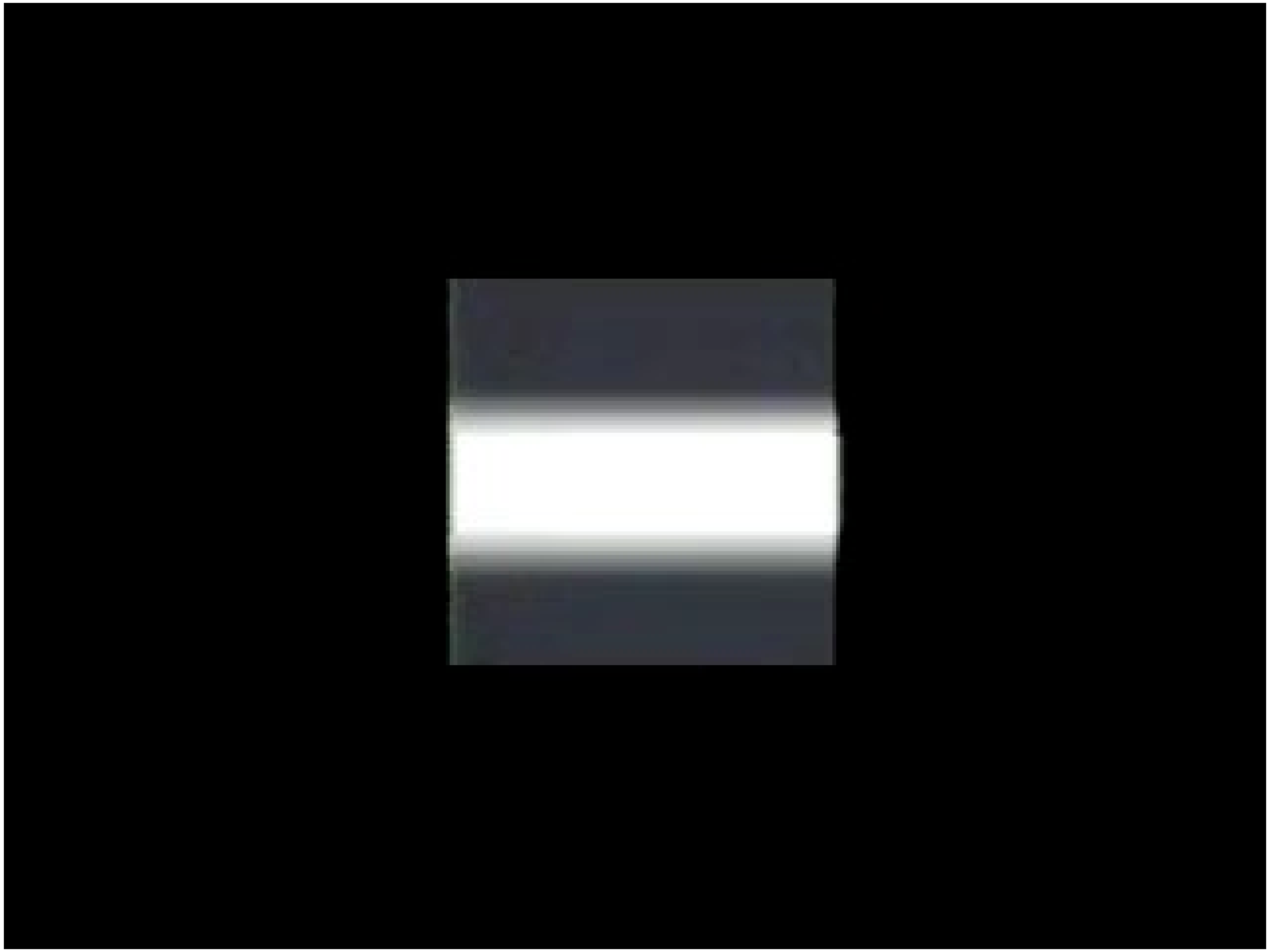
The aperture problem

$$I_x u + I_y v + I_t = 0 \Rightarrow \left(I_x, I_y\right) \cdot \left(u, v\right) = -I_t$$

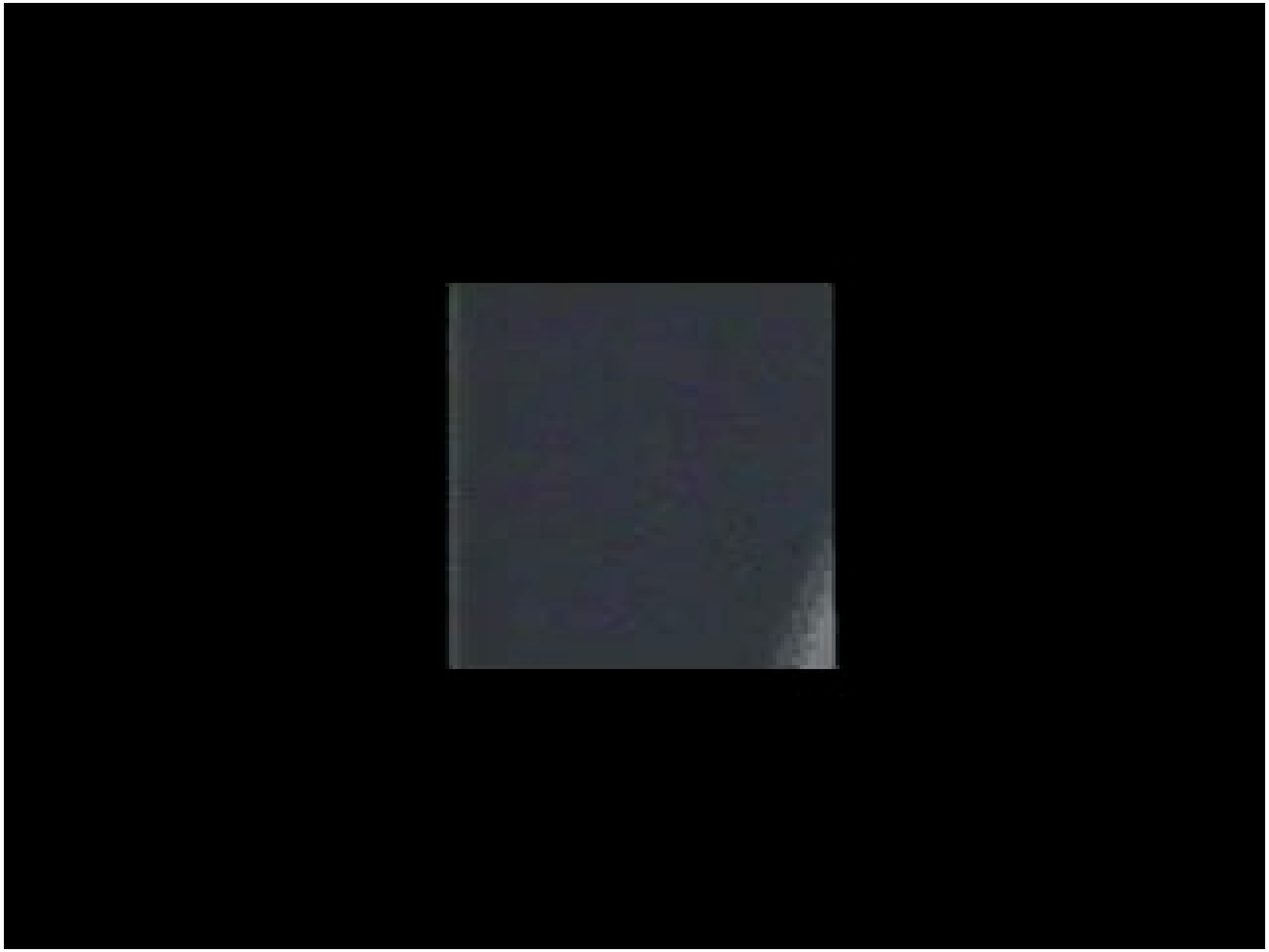
Aperture problem : only the component along the gradient can be retrieved

$$\frac{I_t}{\sqrt{I_x^2 + I_y^2}}$$

The aperture problem



Remarks



Remarks

1. The underdetermined nature could be solved using higher derivatives of intensity

2. for some intensity patterns, e.g. patches with a planar intensity profile, the aperture problem cannot be resolved anyway.

For many images, large parts have planar intensity profiles… higher-order derivatives than 1st order are typically not used (also because they are noisy)

Horn & Schunck algorithm

Breaking the spell via an …
additional smoothness constraint :

$$e_s = \iint ((u_x^2 + u_y^2) + (v_x^2 + v_y^2))\,dxdy,$$

to be minimized,
besides the OF constraint equation term

$$e_c = \iint (I_x u + I_y v + I_t)^2\,dxdy,$$

The integrals are over the image.

Horn & Schunck algorithm

Breaking the spell via an …
additional smoothness constraint :

$$e_s = \iint ((u_x^2 + u_y^2) + (v_x^2 + v_y^2))dxdy,$$

to be minimized,
besides the OF constraint equation term

$$e_c = \iint (I_x u + I_y v + I_t)^2 dxdy,$$

minimize $e_s + \lambda e_c$ (also reduces influence of noise)

The calculus of variations

look for functions that extremize *functionals*

(a functional is a function that takes a vector as its input argument, and returns a scalar)

like for our functional:

$$\iint ((u_x^2 + u_y^2) + (v_x^2 + v_y^2))\,dxdy$$

$$+ \lambda \iint (I_x u + I_y v + I_t)^2\,dxdy$$

what are the optimal *u(x,y)* and *v(x,y)* ?

Computer Vision

The calculus of variations

look for functions that extremize *functionals*

$$I = \int_{x_1}^{x_2} F(x, f, f')\,dx \quad \text{with } f = f(x),\; f' = \frac{df}{dx}$$

$$f(x_1) = f_1 \quad \text{and} \quad f(x_2) = f_2$$

Calculus of variations

Suppose

1. $f(x)$ is a solution

2. $\eta(x)$ is a test function with $\eta(x_1) = 0$ and $\eta(x_2) = 0$

We then consider

$$I = \int_{x_1}^{x_2} F(x, f, f')\,dx \quad \text{with } f = f(x),\ f' = \frac{df}{dx}$$

$$f \rightarrow f + \varepsilon\eta$$

$$I = \int_{x_1}^{x_2} F(x, f + \varepsilon\eta\,,\ f' + \varepsilon\eta')\,dx$$

Rationale: supposed f is the solution, then any deviation should result in a worse I; when applying classical optimization over the values of ε the optimum should be $\varepsilon = 0$

Computer Vision

Calculus of variations

Suppose

1. $f(x)$ is a solution

2. $\eta(x)$ is a test function with $\eta(x_1) = 0$ and $\eta(x_2) = 0$

We then consider

$$I = \int_{x_1}^{x_2} F(x, f + \varepsilon\eta\,,\ f' + \varepsilon\eta')\,dx$$

With this trick, we reformulate an optimization over a function into a classical optimization over a scalar… a problem we know how to solve

Computer Vision

Calculus of variations

Suppose

1. $f(x)$ is a solution

2. $\eta(x)$ is a test function with $\eta(x_1)= 0$ and $\eta(x_2) = 0$

$$I = \int_{x_1}^{x_2} F(x, f + \varepsilon\eta\,,\ f' + \varepsilon\eta')\,dx$$

for the optimum :

$$\frac{dI}{d\varepsilon}\Big|_{\varepsilon=0} = 0$$

Around the optimum, the derivative should be zero

Computer Vision

Calculus of variations

Suppose

1. $f(x)$ is a solution

2. $\eta(x)$ is a test function with $\eta(x_1) = 0$ and $\eta(x_2) = 0$

$$I = \int_{x_1}^{x_2} F(x, f + \varepsilon\eta \,,\, f' + \varepsilon\eta')\,dx$$

for the optimum :

$$\int_{x_1}^{x_2} (\eta(x) F_f + \eta'(x) F_{f'})\,dx = 0$$

$f + \varepsilon\eta$ with $\varepsilon = 0$ $\quad$ $f' + \varepsilon\eta'$ with $\varepsilon = 0$

Calculus of variations

$$\int_{x_1}^{x_2} (\eta(x)F_f + \eta'(x)F_{f'})dx = 0$$

Using integration by parts:

$$\int_{x_1}^{x_2} \frac{d}{dx}(g\,h)\,dx = \int_{x_1}^{x_2} (\frac{dg}{dx}h + \frac{dh}{dx}g)dx = [gh]_{x_1}^{x_2}$$

where

$$[gh]_{x_1}^{x_2} = g(x_2)h(x_2) - g(x_1)h(x_1)$$

Computer Vision

Calculus of variations

$$\int_{x_1}^{x_2} (\eta(x)F_f + \eta'(x)F_{f'})dx = 0$$

Using integration by parts $\int_{x_1}^{x_2} \frac{d}{dx}(\eta(x)F_{f'})\,dx$:

$$\int_{x_1}^{x_2} \eta'(x)F_{f'} + \eta(x)\frac{d}{dx}F_{f'}dx = \left[\eta(x)F_{f'}\right]_{x_1}^{x_2}$$

Computer Vision

Calculus of variations

$$\int_{x_1}^{x_2} (\eta(x)F_f + \eta'(x)F_{f'})dx = 0$$

Using integration by parts $\int_{x_1}^{x_2} \frac{d}{dx}(\eta(x)F_{f'})\, dx$:

$$\int_{x_1}^{x_2} \eta'(x)F_{f'} + \eta(x)\frac{d}{dx}F_{f'}dx = \left[\eta(x)F_{f'}\right]_{x_1}^{x_2}$$

Calculus of variations

Computer Vision

$$\int_{x_1}^{x_2} (\eta(x)F_f + \eta'(x)F_{f'})dx = 0$$

Using integration by parts $\int_{x_1}^{x_2} \frac{d}{dx}(\eta(x)F_{f'})\, dx$:

$$\int_{x_1}^{x_2} \eta'(x)F_{f'}dx = \left[\eta(x)F_{f'}\right]_{x_1}^{x_2} - \int_{x_1}^{x_2} \eta(x)\frac{d}{dx}F_{f'}\, dx,$$

Calculus of variations

$$\int_{x_1}^{x_2} (\eta(x)F_f + \eta'(x)F_{f'})dx = 0$$

Using integration by parts $\int_{x_1}^{x_2} \frac{d}{dx}(\eta(x)F_{f'})\,dx$:

$$\int_{x_1}^{x_2} \eta'(x)F_{f'}dx = \qquad - \int_{x_1}^{x_2} \eta(x)\frac{d}{dx}F_{f'}\,dx,$$

Therefore

$$\int_{x_1}^{x_2} \eta(x)(F_f - \frac{d}{dx}F_{f'})dx = 0$$

regardless of $\eta(x)$, then $F_f - \frac{d}{dx}F_{f'} = 0$

Euler-Lagrange equation

Computer Vision

Calculus of variations

Generalizations

- 1. $I = \int_{x_1}^{x_2} F(x, f_1, f_2, \ldots, f_1', f_2', \ldots) dx$

  Simultaneous Euler-Lagrange equations, i.c. one for *u* and one for *v* :

  $$F_{fi} - \frac{d}{dx} F_{f_i'} = 0$$

Calculus of variations

Generalizations

- 1. $I = \int_{x_1}^{x_2} F(x, f_1, f_2, \ldots, f_1', f_2', \ldots) dx$

  Simultaneous Euler-Lagrange equations, i.c. one for *u* and one for *v* :

  $$F_{fi} - \frac{d}{dx} F_{f_i'} = 0$$

  As we add $\varepsilon_1 \ \eta_1$ to f_1 , and $\varepsilon_2 \ \eta_2$ to f_2
  then repeat, once deriving w.r.t. ε_1 ,
  once w.r.t. ε_2
  thus obtaining a system of 2 PDEs

Calculus of variations

Generalizations

- 1. $$I = \int_{x_1}^{x_2} F(x, f_1, f_2, \ldots, f_1', f_2', \ldots) dx$$

  Simultaneous Euler-Lagrange equations, i.c. one for *u* and one for *v* :

  $$F_{fi} - \frac{d}{dx} F_{f_i'} = 0$$

- 2. 2 independent variables x and y

$$I = \int\int_D F(x, y, f + \varepsilon\eta, f_x + \varepsilon\eta_x, f_y + \varepsilon\eta_y) dxdy$$

Calculus of variations

Hence

$$0 = \iint_D (\eta F_f + \eta_x F_{f_x} + \eta_y F_{f_y}) dxdy$$

Now by Gauss's integral theorem,

$$\iint_D \left(\frac{\partial Q}{\partial x} + \frac{\partial P}{\partial y} \right) dxdy = \int_{\partial D} (Q dy - P dx),$$

such that

$$\iint_D \frac{\partial(\eta F_{f_x})}{\partial x} + \frac{\partial(\eta F_{f_y})}{\partial y} dxdy = \int_{\partial D} (\eta\, F_{f_x} dy - \eta\, F_{f_y} dx)$$

$= 0$

Computer Vision

Calculus of variations

$$\int\int_D \frac{\partial(\eta F_{f_x})}{\partial x} + \frac{\partial(\eta F_{f_y})}{\partial y} dxdy = 0$$

$$\int\int_D (\eta_x F_{f_x} + \eta_y F_{f_y})\, dxdy + \int\int_D (\eta \frac{\partial F_{f_x}}{\partial x} + \eta \frac{\partial F_{f_y}}{\partial y})\, dxdy = 0$$

Computer Vision

Calculus of variations

$$0=\int\int_D(\eta F_f+\eta_x F_{f_x}+\eta_y F_{f_y})dxdy$$

$$\int\int_D \frac{\partial(\eta F_{f_x})}{\partial x}+\frac{\partial(\eta F_{f_y})}{\partial y}dxdy=0$$

$$\int\int_D(\eta_x F_{f_x}+\eta_y F_{f_y})\,dxdy=-\int\int_D \eta\left(\frac{\partial F_{f_x}}{\partial x}+\frac{\partial F_{f_y}}{\partial y}\right)dxdy$$

Consequently,

$$0=\int\int_D \eta\left(F_f-\frac{\partial}{\partial x}F_{f_x}-\frac{\partial}{\partial y}F_{f_y}\right)dxdy$$

for all test functions η , thus

$$F_f-\frac{\partial}{\partial x}F_{f_x}-\frac{\partial}{\partial y}F_{f_y}=0$$

is the Euler-Lagrange equation

Computer Vision

Horn & Schunck

The Euler-Lagrange equations :

$$F_u - \frac{\partial}{\partial x} F_{u_x} - \frac{\partial}{\partial y} F_{u_y} = 0$$

$$F_v - \frac{\partial}{\partial x} F_{v_x} - \frac{\partial}{\partial y} F_{v_y} = 0$$

In our case ,

$$F = (u_x^2 + u_y^2) + (v_x^2 + v_y^2) + \lambda (I_x u + I_y v + I_t)^2,$$

so the Euler-Lagrange equations are

$$\Delta u = \lambda (I_x u + I_y v + I_t) I_x,$$

$$\Delta v = \lambda (I_x u + I_y v + I_t) I_y,$$

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$ is the Laplacian operator

Computer Vision

Horn & Schunck

Remarks :

1. Coupled PDEs solved using iterative methods and finite differences (iteration i)

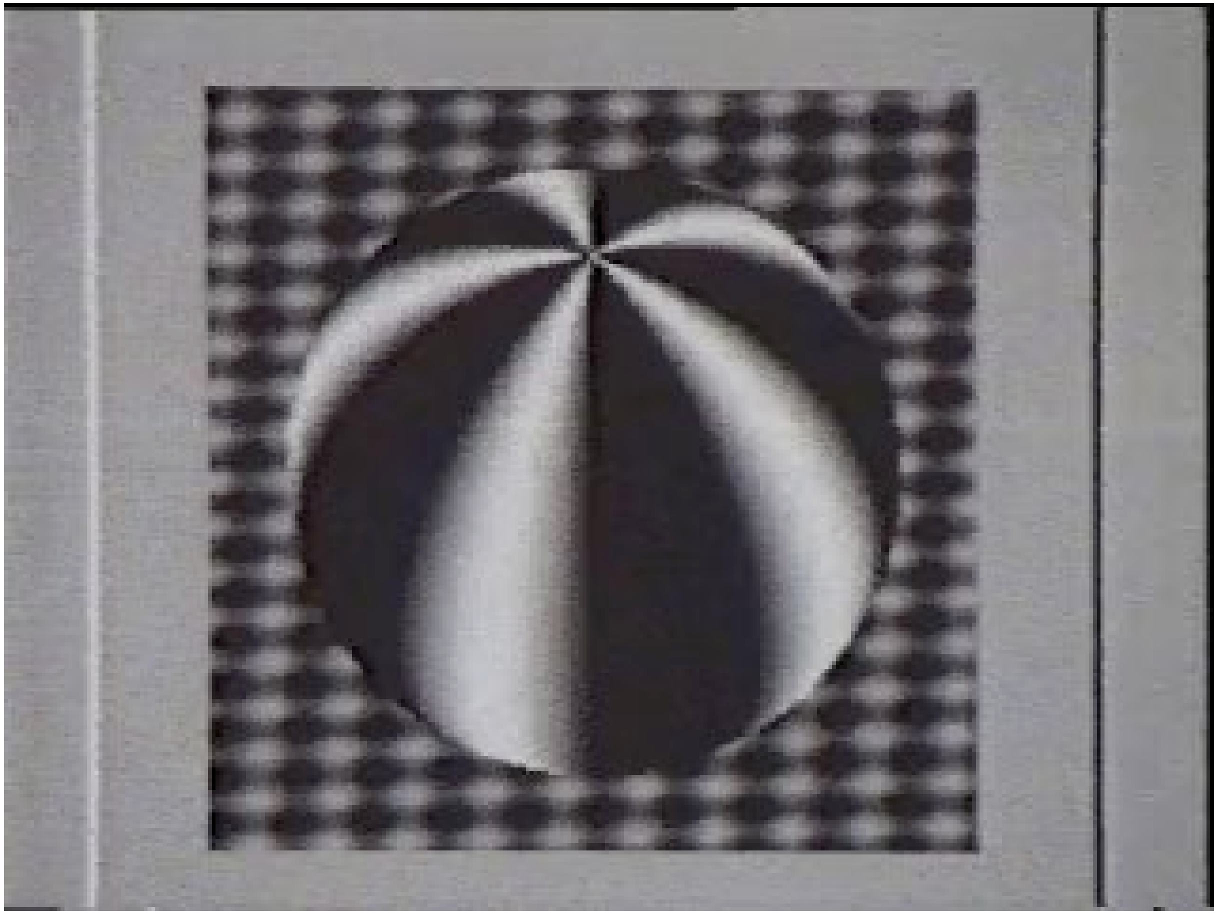
$$\frac{\partial u}{\partial i} = \Delta u - \lambda(I_x u + I_y v + I_t) I_x,$$

$$\frac{\partial v}{\partial i} = \Delta v - \lambda(I_x u + I_y v + I_t) I_y,$$

2. More than two frames allow for a better estimation of I_t

3. Information spreads from edge- and corner-type patterns

Horn & Schunck, example result



Horn & Schunck, remarks

1. Errors at object boundaries

   (where the smoothness constraint is no longer valid)

2. Example of *regularisation*

   (selection principle for the solution of ill-posed problems by imposing an extra generic constraint, like here smoothness)

condensation filter

condensation filter

as an example of a `tracker', shifting the emphasis from pixels to objects…

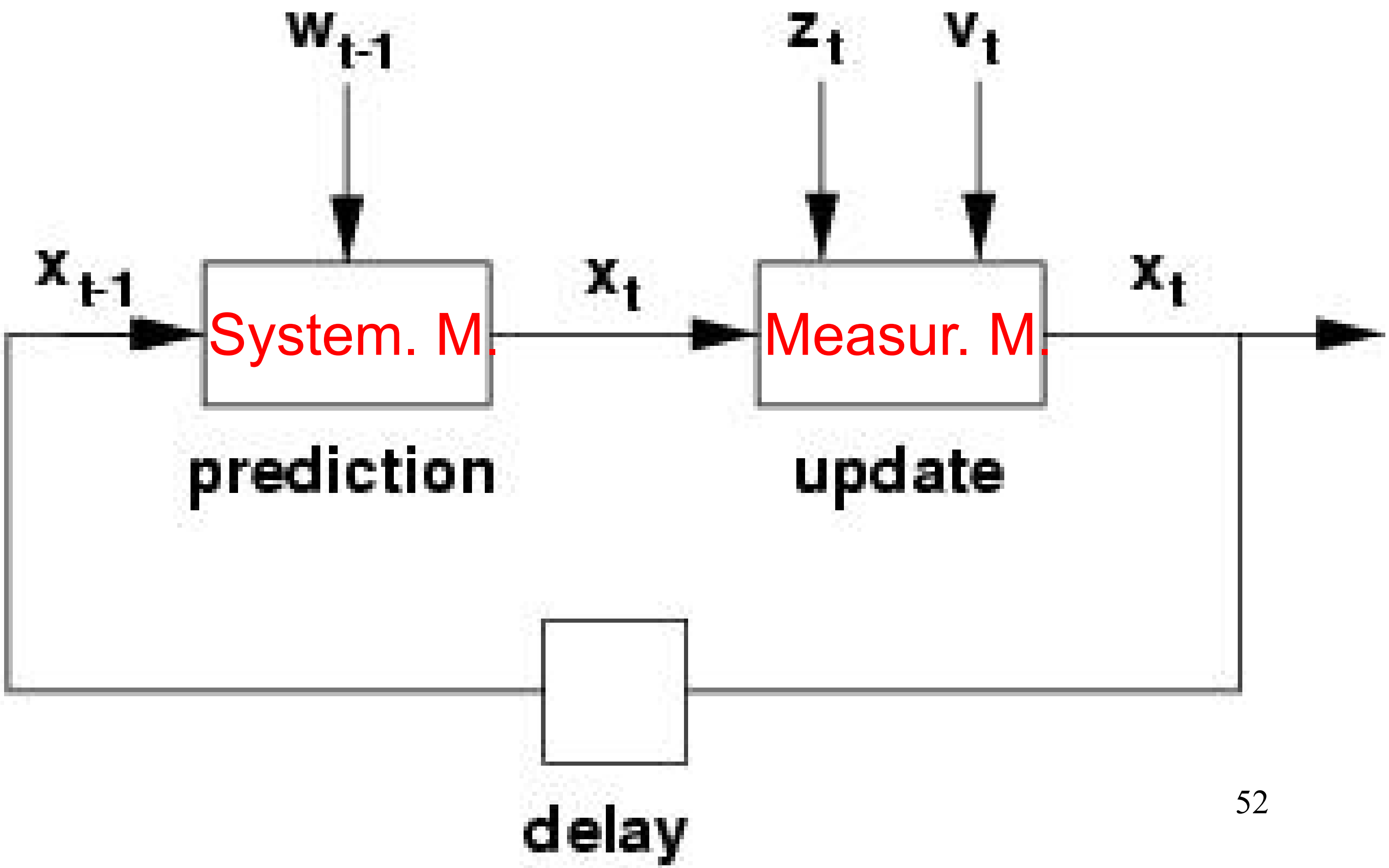
Condensation tracker

x_t state vector

z_t observation vector

w_t noise in system model

v_t noise in measurement model



Condensation tracker

1. Prediction , based on the *system model*

$$x_t = f_{t-1}(x_{t-1}, w_{t-1})$$

(f = system transition function)

2. Update , based on the *measurement model*

$$z_t = h_t(x_t, v_t)$$

(h = measurement function)

$Z_t = (z_1, \ \ldots \ , z_t)$ is the *history* of observations

Condensation tracker

Example

dots indicate time derivatives

System model

$$x_t = (p_t, \dot{p}_t)$$

$$p_t = p_{t-1} + \Delta t\ \dot{p}_{t-1} + w_{p,t-1}$$ position

$$\dot{p}_t = \dot{p}_{t-1} + w_{\dot{p},t-1}$$ velocity

Measurement model

$$z_t = p_t + v_t$$

Condensation tracker

Recursive Bayesian filter

Object not as a single state but a prob. distribution

Computer Vision

Condensation tracker

Recursive Bayesian filter

Object not as a single state but a prob. Distribution
(p here means probability…)

1. Prediction

$$p(x_t \mid Z_{t-1}) = \int p(x_t \mid x_{t-1})\, p(x_{t-1} \mid Z_{t-1})\, dx_{t-1}$$

2. Update

$$p(x_t \mid Z_t) = \frac{p(z_t \mid x_t)\, p(x_t \mid Z_{t-1})}{p(z_t \mid Z_{t-1})}$$

Condensation tracker

Recursive Bayesian filter

Object not as a single state but a prob. distribution

1. Prediction

$$p(x_t \mid Z_{t-1}) = \int p(x_t \mid x_{t-1}) \, p(x_{t-1} \mid Z_{t-1}) \, dx_{t-1}$$

2. Update

$$p(x_t \mid Z_t) = \frac{p(z_t \mid x_t) \, p(x_t \mid Z_{t-1})}{p(z_t \mid Z_{t-1})}$$

$p(z_t \mid Z_{t-1})$ can be considered a normalization factor

Computer Vision

Condensation tracker

Recursive Bayesian filter

Object not as a single state but a prob. distribution

Bayes´ rule

$$p(b \mid a)\, p(a) = p(a \mid b)\, p(b) = p(a,b)$$

here $p(x_t, z_t \mid Z_{t-1})$

2. Update

$$p(x_t \mid Z_t) = \frac{p(z_t \mid x_t)\, p(x_t \mid Z_{t-1})}{p(z_t \mid Z_{t-1})}$$

$p(z_t \mid Z_{t-1})$ can be considered a normalization factor

Condensation tracker

Recursive Bayesian filter

Object not as a single state but a prob. distribution

Bayes´ rule

$$p(x_t, z_t \mid Z_{t-1}) = p(x_t \mid Z_t)\, p(z_t \mid Z_{t-1}) = p(z_t \mid x_t)\; p(x_t \mid Z_{t-1})$$

2. Update

$$p(x_t \mid Z_t) = \frac{p(z_t \mid x_t)\; p(x_t \mid Z_{t-1})}{p(z_t \mid Z_{t-1})}$$

$p(z_t \mid Z_{t-1})$ can be considered a normalization factor

Computer Vision

Condensation tracker

Recursive Bayesian filter

Object not as a single state but a prob. distribution

1. Prediction

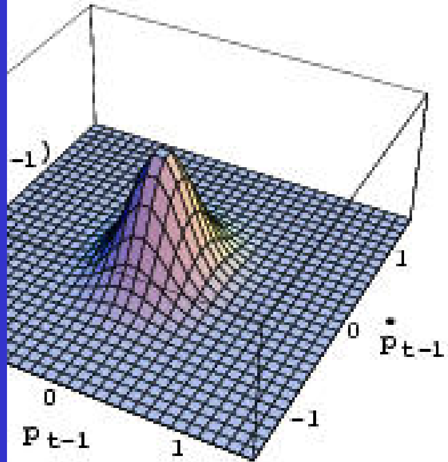
$$p(x_t \mid Z_{t-1}) = \int p(x_t \mid x_{t-1}) \, p(x_{t-1} \mid Z_{t-1}) \, dx_{t-1}$$

2. Update

$$p(x_t \mid Z_t) = \frac{p(z_t \mid x_t) \, p(x_t \mid Z_{t-1})}{p(z_t \mid Z_{t-1})}$$

Condensation tracker

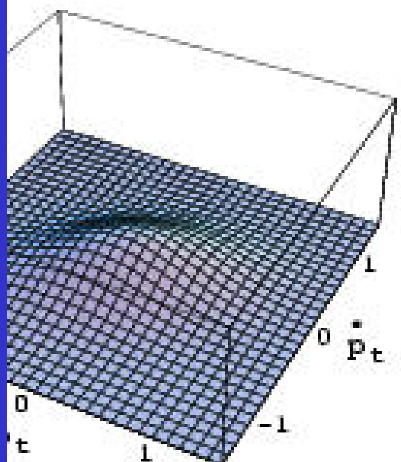
Example cont'd



$$p(p_{t-1}, \dot{p}_{t-1} \mid z_{t-1})$$

a posteriori prob. distr. at *t-1*

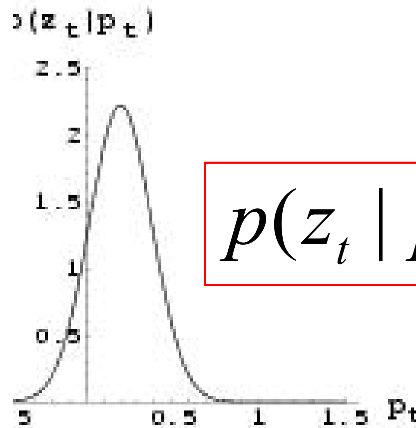
prediction



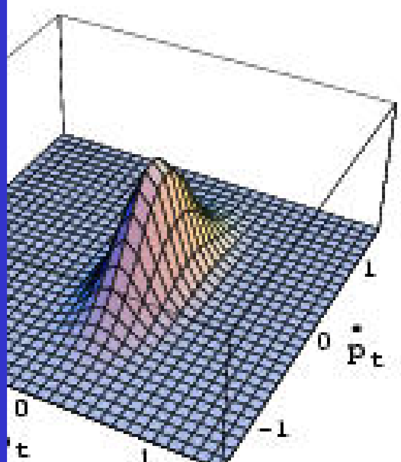
$$p(p_t, \dot{p}_t \mid z_{t-1})$$

a priori prob. distr. at *t*

update



$$p(z_t \mid p_t)$$



$$p(p_t, \dot{p}_t \mid z_t)$$

Condensation tracker

Recursive Bayesian filter

Calculating $p(x_t \mid Z_{t-1}) = \int p(x_t \mid x_{t-1})\, p(x_{t-1} \mid Z_{t-1})\, dx_{t-1}$ numerically is very time consuming, and the prob. distributions have to be known…

Analytic solutions are only available for the simplest of cases, e.g. when distr. are Gaussian and the system and measurement models are linear…
(Kalman filter, 1960 - Kalman was prof. at ETH, D-ITET)

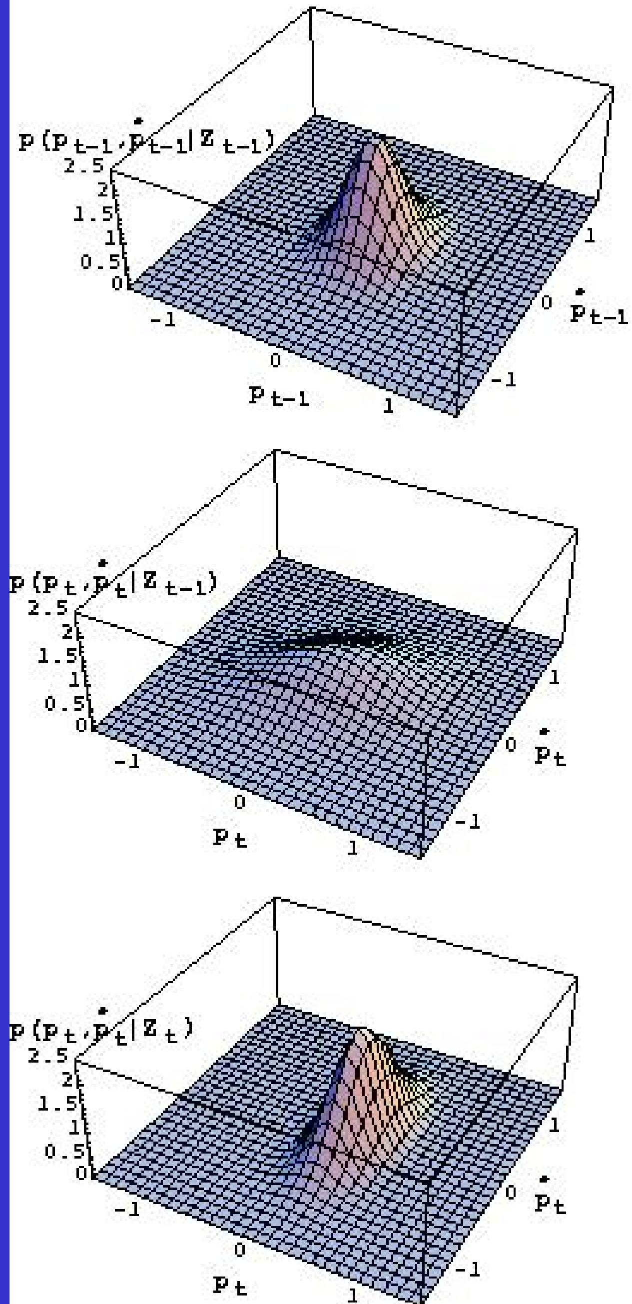
Condensation tracker

Recursive Bayesian filter

Calculating $p(x_t \mid Z_{t-1}) = \int p(x_t \mid x_{t-1})\, p(x_{t-1} \mid Z_{t-1})\, dx_{t-1}$ numerically is very time consuming, and the prob. distributions have to be known…

Analytic solutions are only available for the simplest of cases, e.g. when distr. are Gaussian and the system and measurement models are linear…

That's where CONDENSATION comes in, acronym for CONditional DENSity propagATION



In our example model is linear, distributions Gaussian

System model

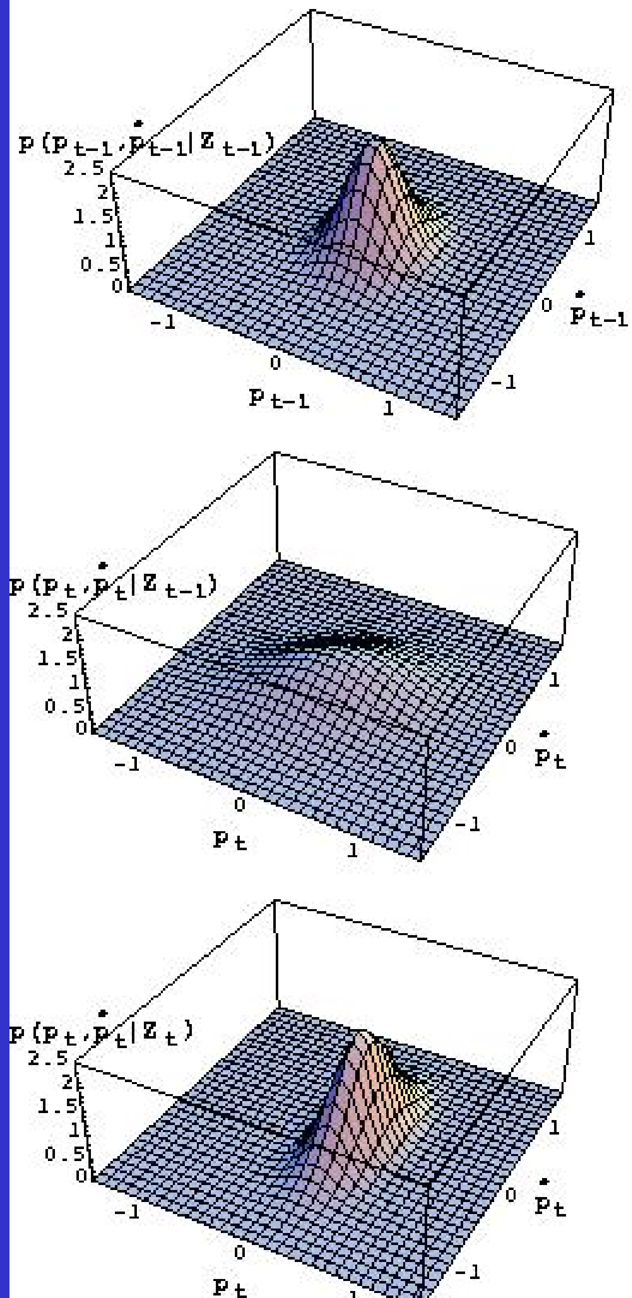
$$p_t = p_{t-1} + \Delta t\ \dot{p}_{t-1} + w_{p,t-1}$$

$$\dot{p}_t = \dot{p}_{t-1} + w_{\dot{p},t-1}$$

Measurement model

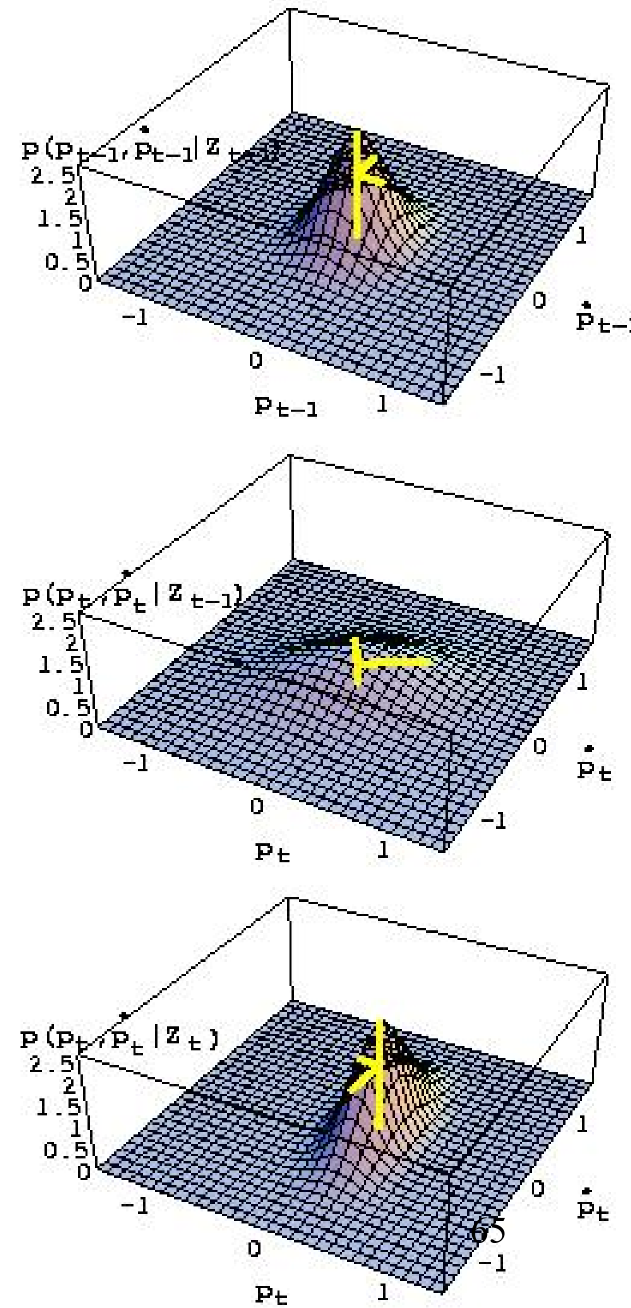
$$z_t = p_t + v_t$$

Computer Vision



K
A
L
M
A
N

F
I
L
T
E
R



Condensation tracker

The probability distribution is represented by a sample set *S* (set of selected states s)

$$S = \left\{ (s^{(n)}, \pi^{(n)}) \mid n = 1 \ldots N \right\}$$

With π a weight determining the sampling probability

Condensation tracker

1. prediction

Start with S_{t-1} , the sample set of the previous step, and apply the system model to each sample, yielding predicted samples $s_t^{'(n)}$

2. update

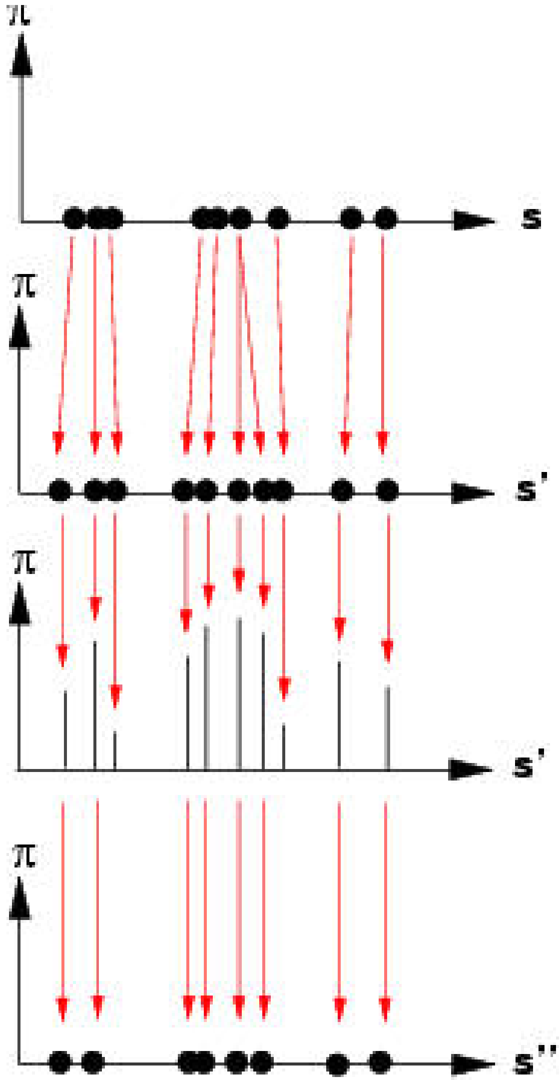
Sample from the predicted set, where samples are drawn with replacement and with probability

$$\pi^{(n)} = p(z_t \mid s_t^{'(n)})$$

(i.e. using meas. model)

In the limit (large N) equivalent to Bayesian tracker

Condensation tracker



$$p(p_{t-1}, \dot{p}_{t-1} \mid z_{t-1})$$

prediction

$$p(p_t, \dot{p}_t \mid z_{t-1})$$

weighing

$$p(z_t \mid p_t)$$

update

$$p(p_t, \dot{p}_t \mid z_t)$$

Condensation tracker

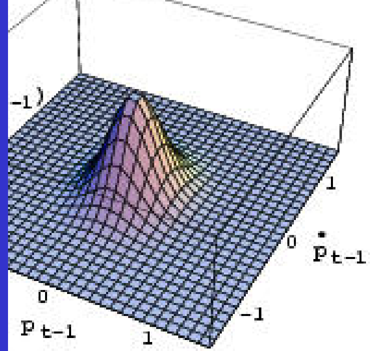
NOTE

Sample may be drawn multiple times, but noise will yield different predictions for samples corresponding to the same state after drawing.

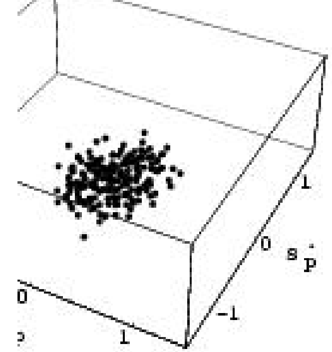
This diversification through noise is important, as otherwise fewer and fewer different samples would survive

Computer Vision

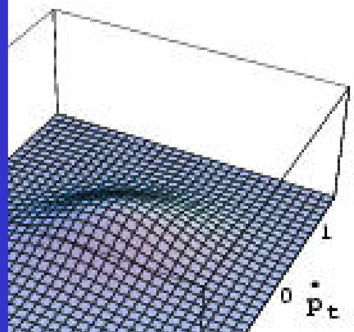
Example cont'd



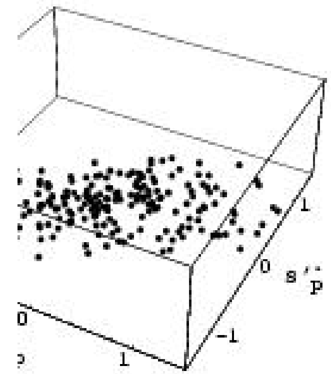
$$p(p_{t-1}, \dot{p}_{t-1} \mid z_{t-1})$$

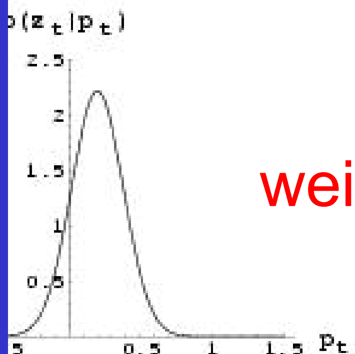


prediction

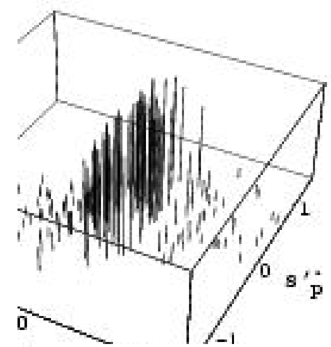


$$p(p_t, \dot{p}_t \mid z_{t-1})$$

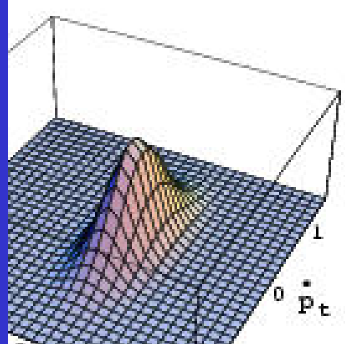




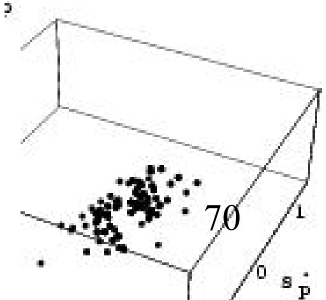
weighing with $p(z_t \mid p_t)$



update



$$p(p_t, \dot{p}_t \mid z_t)$$



C O N D E N S A T I O N

Condensation tracker

Comparison with Kalman filter

Condensation	Kalman-Bucy
Unrestricted system models	Linear system models
Unrestricted noise models	Gaussian noise
Multiple hypotheses	Unimodal
Discretisation error	Exact solution
Postprocessing for interpret.	Direct interpretation

Condensation tracker

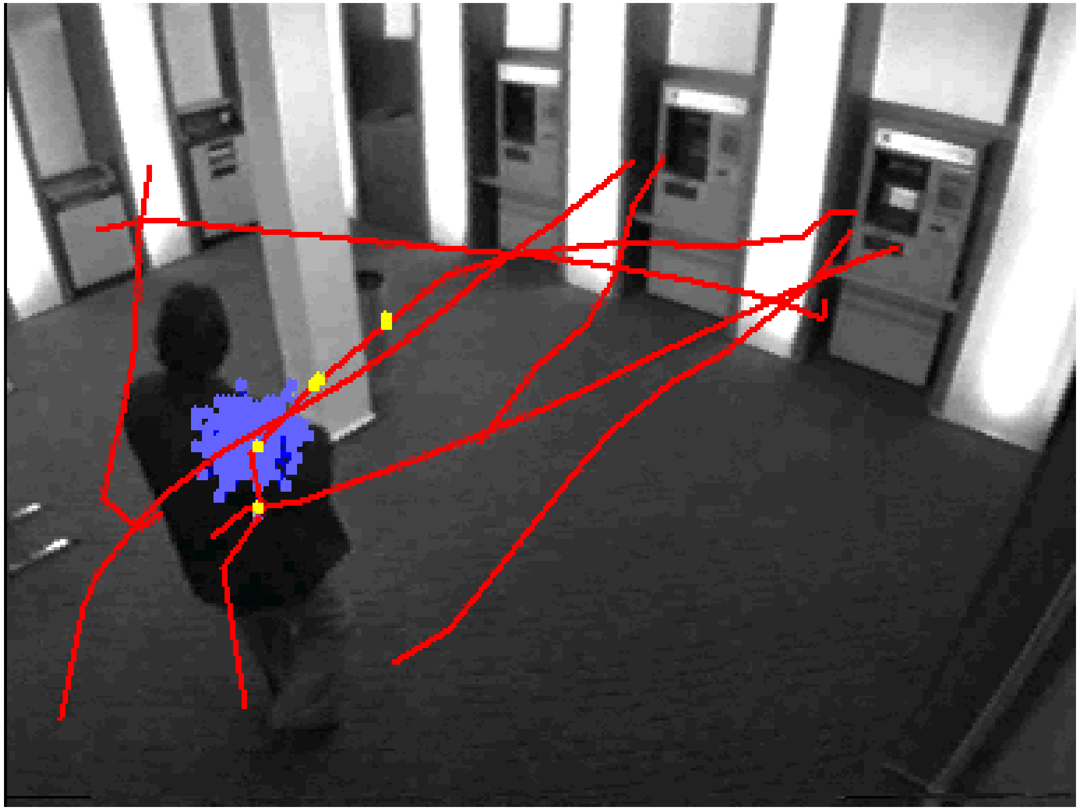
Computer Vision

Condensation tracker

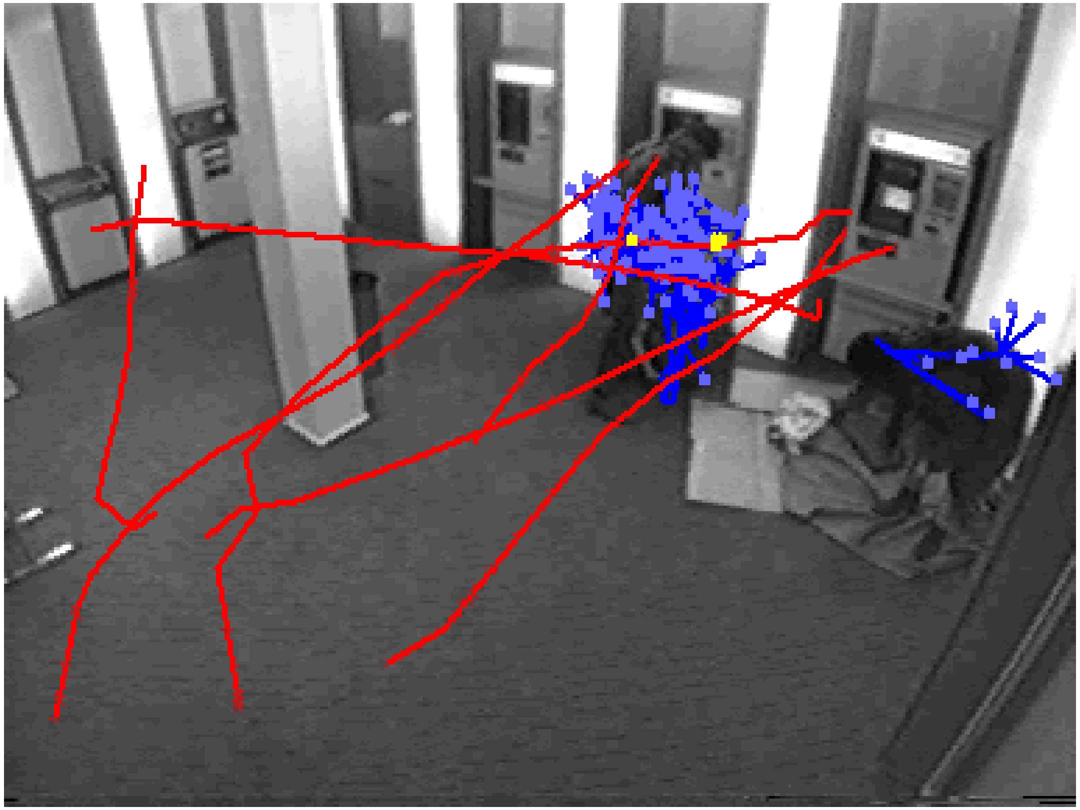
Condensation tracker

Computer Vision

Condensation tracker

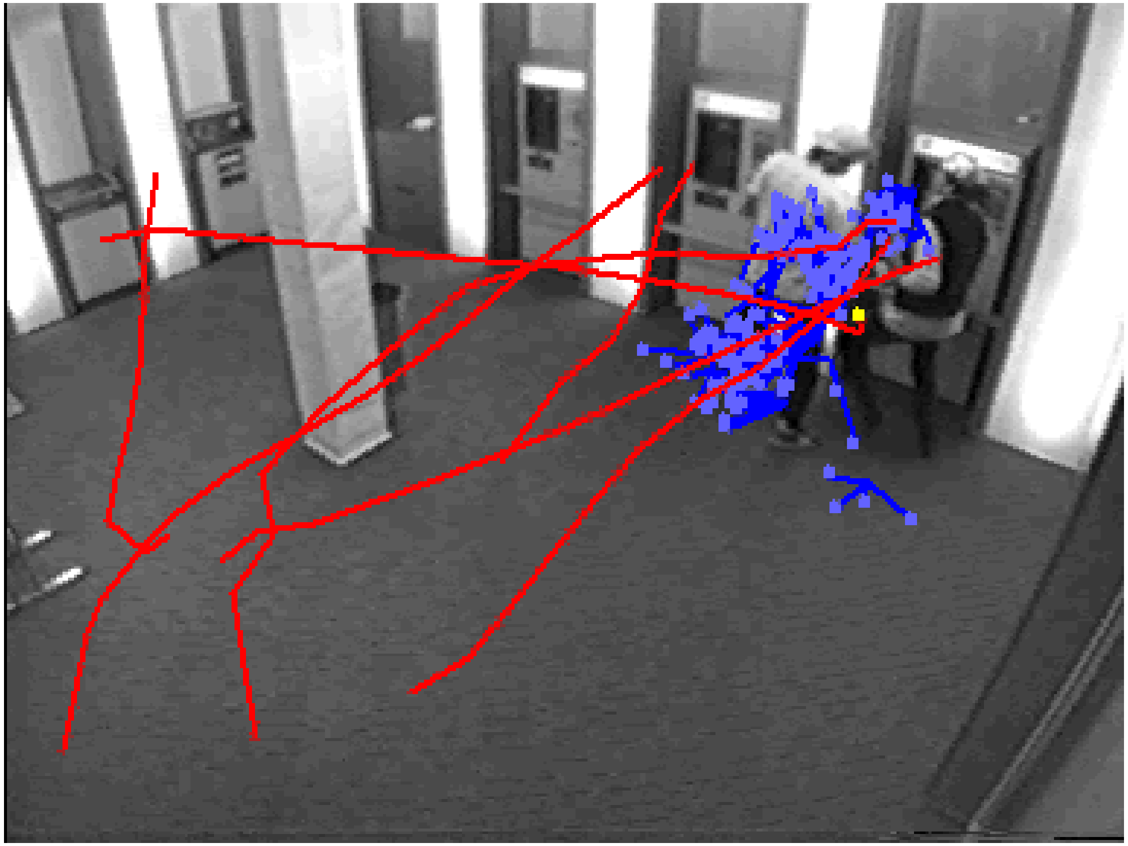


Condensation tracker



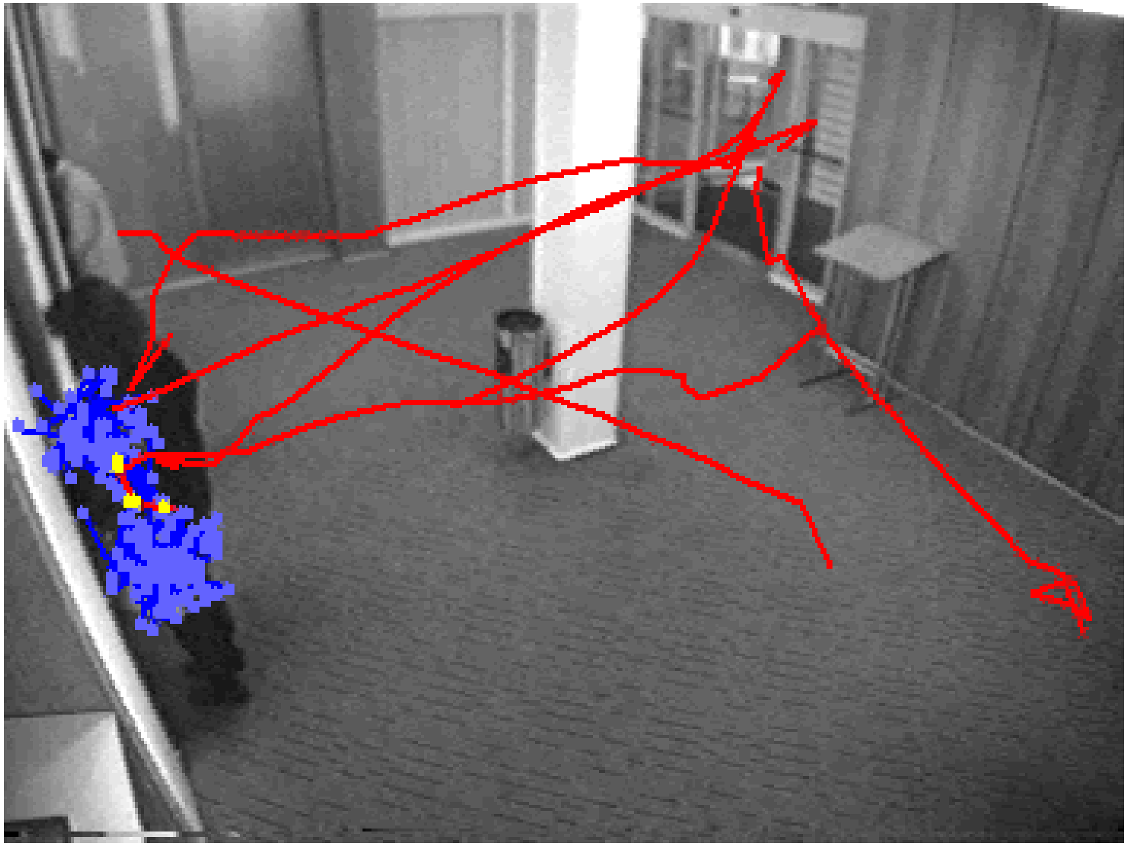
Computer Vision

Condensation tracker



Computer Vision

Condensation tracker



Computer Vision

Computer Vision

Condensation tracker

Elliptical region with prescribed color histogram

System model

$$x_t = x_{t-1} + \Delta t\, \dot{x}_{t-1} + w_{x,t-1}$$
$$y_t = y_{t-1} + \Delta t\, \dot{y}_{t-1} + w_{y,t-1}$$

position

$$\dot{x}_t = \dot{x}_{t-1} + w_{\dot{x},t-1}$$
$$\dot{y}_t = \dot{y}_{t-1} + w_{\dot{y},t-1}$$

velocity

$$H_{x\,t} = H_{x\,t-1} + \Delta t\, \dot{H}_{x\,t-1} + w_{H_x,t-1}$$
$$H_{y\,t} = H_{y\,t-1} + \Delta t\, \dot{H}_{y\,t-1} + w_{H_y,t-1}$$

size

$$\dot{H}_{x\,t} = \dot{H}_{x\,t-1} + w_{\dot{H}_x,t-1}$$
$$\dot{H}_{y\,t} = \dot{H}_{y\,t-1} + w_{\dot{H}_y,t-1}$$

size chance

Condensation tracker

Measurement model

$$\pi = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1-\rho}{2\sigma^2}}$$

with

$$\rho = \sum_{u=1}^{m} \sqrt{p^{(u)} q^{(u)}}$$

where *p* and *q* are the color histograms of a sample and the target, resp.

Condensation tracker

Computer Vision

Mean shift tracker

Computer Vision

Mean shift tracker

Condensation tracker

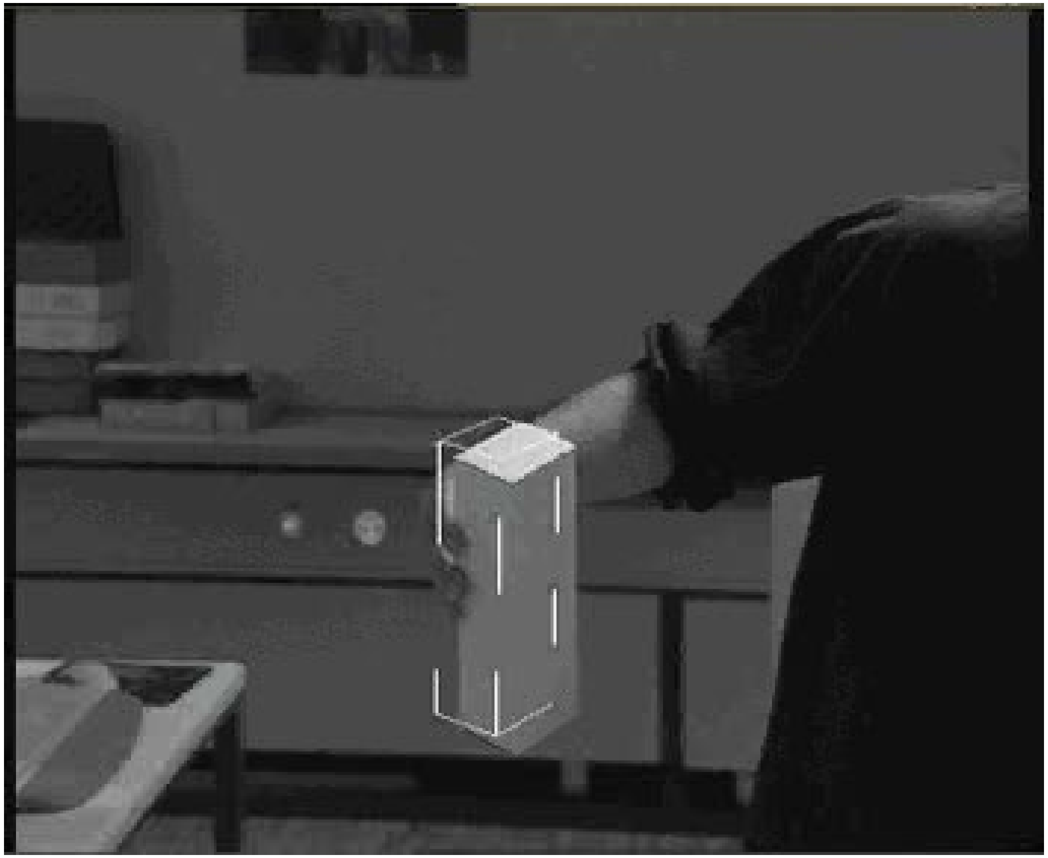
Condensation tracker

Other approaches

1. Model-based tracking (application-specific)
   - active contours (discussed with segmentation)
   - analysis/synthesis schemes

2. Feature tracking (more generic)
   - corner tracking (shown when we discuss 3D)
   - blob/contour tracking
   - intensity profile tracking
   - region tracking

Model-based tracker



(EPFL)

Model-based tracker

(EPFL)

Computer Vision

Motion capture for special effects