ViSP 2.6.0: Visual servoing platform

# ViSP tracking methods overview

October 12th, 2010

Lagadic project

INRIA Rennes-Bretagne Atlantique

http://www.irisa.fr/lagadic

# Tracking methods with ViSP

1. Dot tracker
2. KLT point tracker
3. Moving edges tracker
4. 3D model-based tracker

# 1. Dot tracker

A dot :

- A dot is a part of image where the connected pixels have the same level
- Not necessary an ellipsoid (even it is by default)
- Two classes vpDot and vpDot2

The dot in ViSP is defined by :

- The gray level
- The center of gravity (cog)
- The size
- The moments :
  - The surface $m_{00}$
  - Inertia first order moments along i and j $m_{01}$ and $m_{10}$
  - Inertia first second moments along i and j $m_{02}$ and $m_{20}$
  - $m_{11}$

# 1. Dot tracker with vpDot class

Tracking method : vpDot

- Initialization : Define the dot cog (generally by clicking in the dot)

- Tracking :
  - Binarisation of the image
  - Recursive method to detect all the neighbour components belonging to the object. Start from the previous coordinates of the center of gravity
  - If the dot is found : Compute the parameters (size, moment, …)
  - If no dot is found : The tracking fails

# 1. Dot tracker with vpDot2 class

Tracking method : vpDot2

- Initialization : Define the dot cog (generally by clicking in the dot)

- Tracking :
  - Binarisation of the image
  - From the previous position of the cog, goes right to detect the boundary, then follow the boundary in order to compute the Freeman chain
  - Use the Freeman Chain to find the dot characteristics (cog, size, moments)
  - If a dot is found, check if it looks like the previous dot (size, moment)
  - If no dot or not similar, check if the dot is in an image part around

# 1. Dot Tracker

Advantages :

- Robust : Almost no tracking error if noise and specularity not too strong
- Give information about the tracked objects (cog, moments)
- In vpDot2 : automatic dot detection for initialization and if a dot is lost search a similar dot in a larger ROI

Limits :

- Speed depends on the size : may be slow if the object is big, especially with vpDot.
- vpDot can not track an object if the displacement is too large
- Due to the recursivity limitation on some OS like windows, vpDot is not able to track huge dots.

# 2. KLT point tracker

KLT : Kanade – Lucas -Tomasi

- The goal is to align a template *T(x)* to an input image *I(x)*
- Could be also a small window in the image
- Based on a gradient method

KLT in ViSP :

- vpKltOpencv class that interfaces the KLT implemented in OpenCV
- A patch is defined by :
  - the tracked points in the current image
  - the tracked points in the previous image
- The points lost during the tracking are given if necessary

# 2. KLT point tracker

Tracking method :

- The goal is to move the patch until minimizing the image dissimilarity

$$\sum_x \left[I(W(x,p) - T(x)\right]^2$$

- Where W(x,p) corresponds to a warp which can be more or less complex

For a translation : $W(x,p) = \begin{bmatrix} x + p_1 \\ y + p_2 \end{bmatrix}$

- Assuming that p is known and best increment Δp is sought, the problem becomes :

$$\sum_x \left[I(W(x, p + \Delta p) - T(x)\right]^2$$

# 2. KLT point tracker

Tracking method :

- The problem is linearized by performing first order Taylor extension

$$\sum_x \left[ I(W(x, p + \Delta p)) + \nabla I \frac{\partial W}{\partial p} \Delta p - T(x) \right]^2$$

- The function is derived and set equal to 0 to find the minimum

$$\Delta p = H^{-1} \left[ \nabla I \frac{\partial W}{\partial p} \right]^{\mathrm{T}} \left[ T(x) - I(W(x, p)) \right] \qquad \text{where } \; H = \sum_x \left[ \nabla I \frac{\partial W}{\partial p} \right]^{\mathrm{T}} \left[ \nabla I \frac{\partial W}{\partial p} \right]$$

- p is updated with this method p = p+Δp until Δp < ε

# 2. KLT point tracker

Good features :

- A good point to track :
  - Textured
  - High intensity variations in both x and y axis
- Harris points are used

Advantages :

- Very fast method
- In ViSP, Harris points detection is automatic

Limits :

- Displacement between two images must be small
- In ViSP : use IplImage instead of vpImage : need conversion
- Few parameters are available.

# 3. Moving edges tracker

Moving edges :

- Based on edge detector with gradient filter
- 3 types : line, ellipse and nurbs

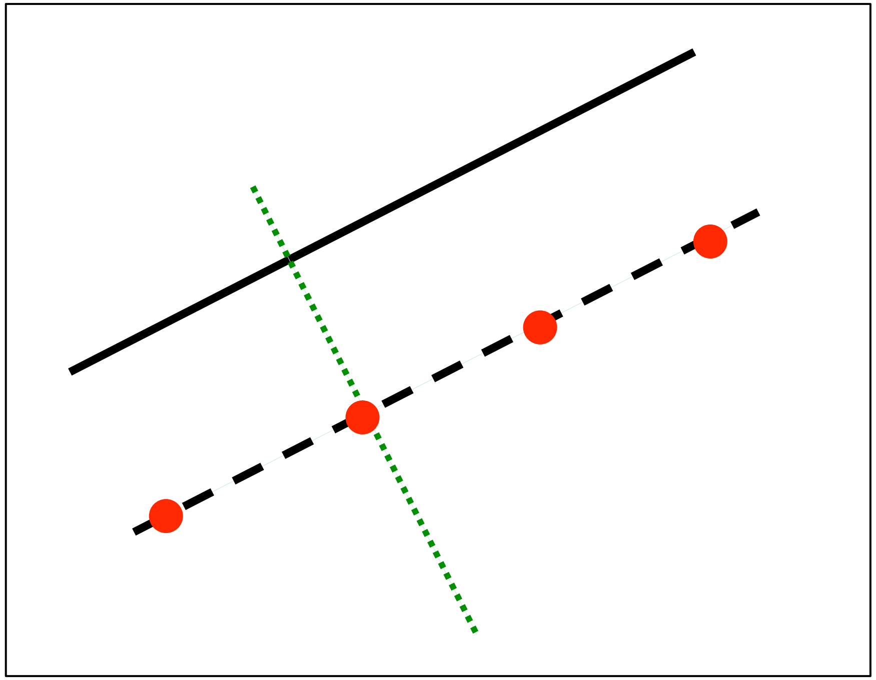
In ViSP:

- vpMeLine, vpMeEllipse, vpMeNurbs classes that inherit from vpMeTracker
- vpMeTracker contains a list of vpMeSite
- Each vpMeSite corresponds to one edge point in the image.
- vpMeSite is defined by :
  - A position (i,j)
  - An angle which corresponds to the normal to the edge
  - An history of the previous convolution result

# 3. Moving edges tracker for a line

Method:

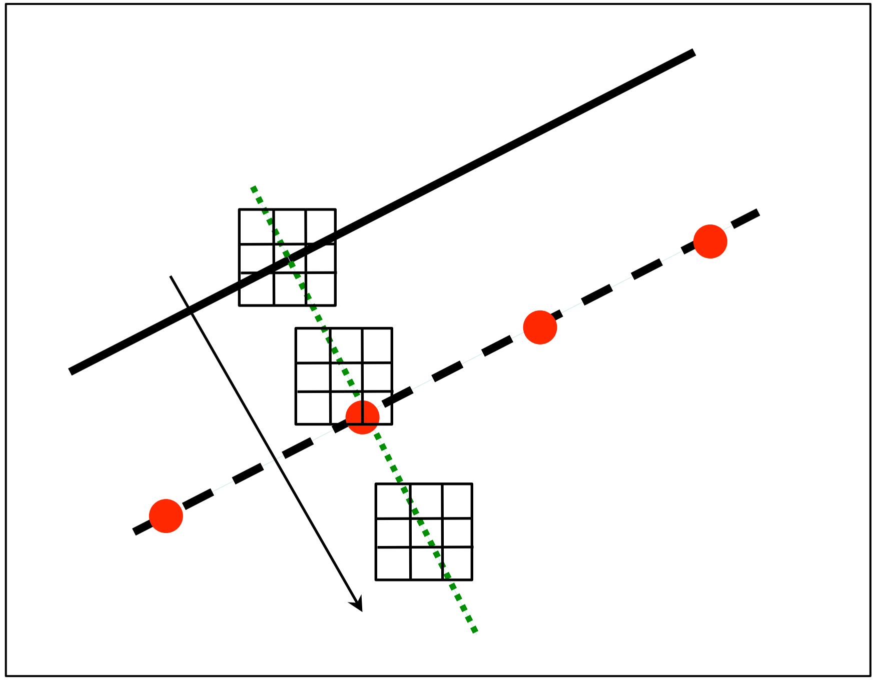
- Capture a new image
- For each vpMeSite : build a list of points along the normal to the edge centered on the edge point previous location



# 3. Moving edges tracker

Method :

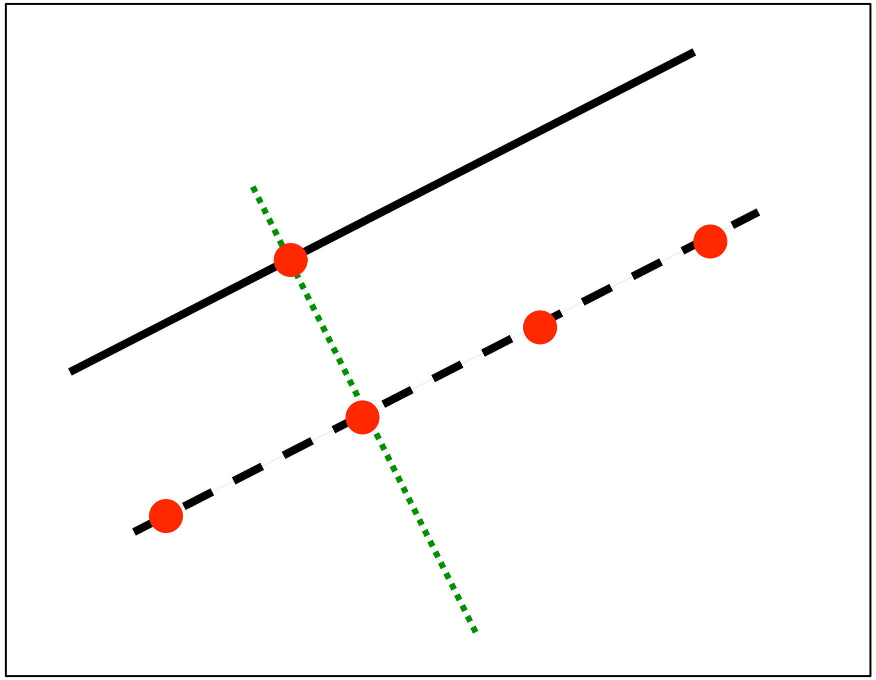
- For each point computes the convolution with a filter optimized to detect edges with an angle near the previous angle



# 3. Moving edges tracker

Method :

- If one point respects the two conditions :
  - The convolution result is close to the previous one
  - The convolution result is high enough
- Then it is considered as the new edge point



# 3. Moving edges tracker

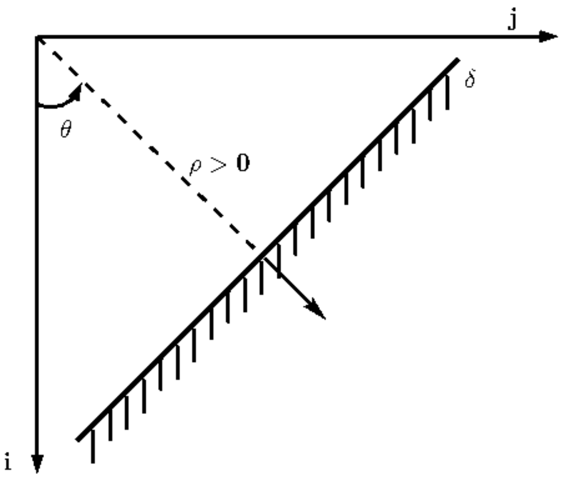
Method :

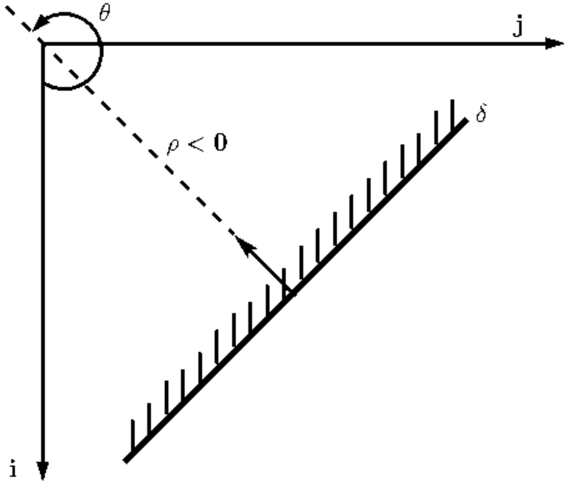
- After all vpMeSite are detected, characteristics of the line, ellipse and nurbs are used to detect outliers
- To suppress outliers : a robust method based on M-Estimators is used

vpMeLine class:

- A line is defined by it's equation

$$ai + bj + c = 0 \qquad i\cos(\theta) + j\sin(\theta) - \rho = 0$$





# 3. Moving edges tracker

vpMeLine class:

- The parameters :
  - a, b and c
  - rho and theta

vpMeEllipse class:

- An ellipse is defined by it’s ellipse equation

$$i^2 + K_0 j^2 + K_1 ij + 2K_2 i + 2K_3 j + K_4 = 0$$

- The K parameters are available in vpMeEllipse class

# 3. Moving edges tracker

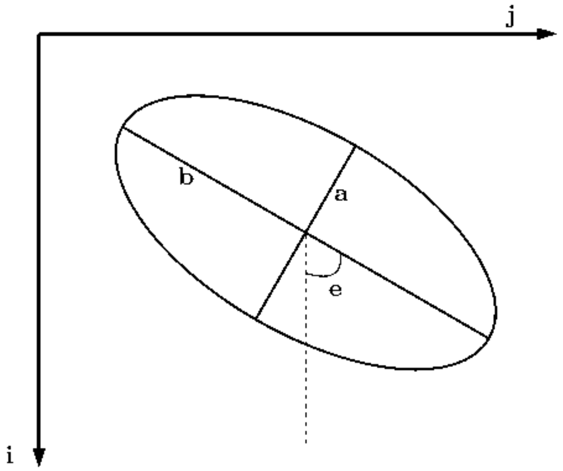
vpMeEllipse class:

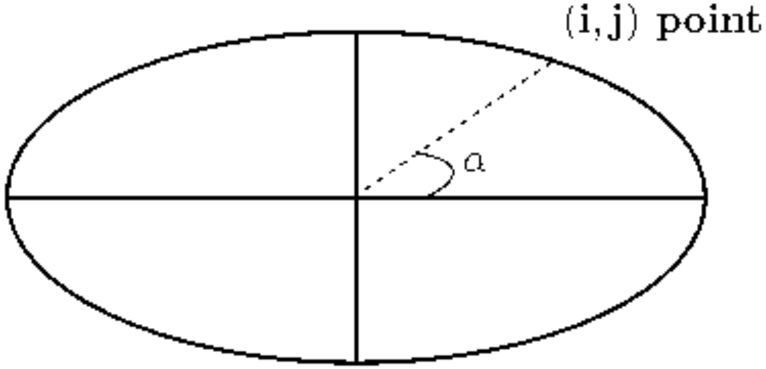
- These two equations describe the ellipse points too :

$$i = i_c + b\cos(e)\cos(\alpha) - a\sin(e)\sin(\alpha)$$

$$j = j_c + b\sin(e)\cos(\alpha) - a\cos(e)\sin(\alpha)$$

- Parameters $i_c$, $j_c$ a, b and e are available in vpMeEllipse.  α is in [0,2π]





# 3. Moving edges tracker

vpMeNurbs class:

- The edge is defined by a parametric curve

$$N_{i,0}(u) = \begin{cases} 1 & \text{si } u_i \le u < u_{i+1} \\ 0 & \text{sinon} \end{cases}$$

where $0 \le u \le 1$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u)$$

Edge points coordinates → $C(u) = \dfrac{\sum_{i=0}^{n} N_{i,p}(u)\omega_i P_i}{\sum_{i=0}^{n} N_{i,p}(u)\omega_i}$

where $P_i$ are control points and $\omega_i$ are weights.

- All the points coordinates are given
- All the derivatives at any points are given too
- All the parameters $N_{i,p}$ $P_i$ and $\omega_i$ are available

# 3. Moving edges tracker

Advantages :

- Gives the equation of the tracked edges
- Fast tracking method
- Useful to initialize visual servoing features implemented in ViSP

Limits :

- The speed depends on the number of points and the size of the search range. If the parameters are not optimal, the algorithm can be “slow”
- vpMeNurbs parameters are difficult to set correctly
- vpMeNurbs is not so robust if the shape of the edge is too complex

# 4. 3D model-based tracker

Model-based tracking :

- Track a 3D model thanks to the moving edges method
- Use a virtual visual servoing
- In ViSP implemented in vpMbEdgeTracker class

Method : Initialization

- Require a 3D model (CAO, WRL, …)
- Need to compute the initial pose
- The pose is used to project the model on the image
- The moving edges points can be initialized

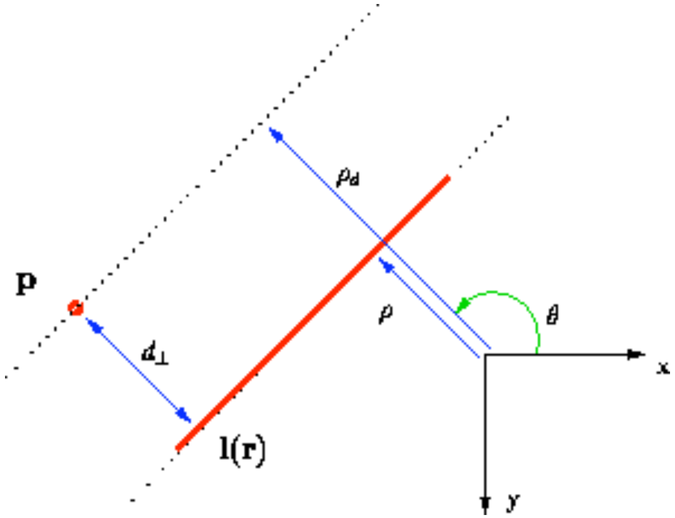
# 4. 3D model-based tracker

Method : Tracking

- Assuming that the pose corresponding to the previous image is known.
- The new lines are tracked
- The goal is to "move" the pose to match the object in the new image with the projection of the model
- The pose to compute is defined by :

$$ {}^{c}\widehat{\mathbf{M}}_o = \arg \min_{{}^{c}R_o, {}^{c}t_o} \sum_i \left( \mathbf{p}_{d_i} - pr({}^{c}\widehat{\mathbf{M}}_o {}^{o}\mathbf{P}_i) \right)^2 $$

- The entire model is taken into account during the minimization



p
$\rho_d$
$\rho$
$\theta$
$d_\perp$
x
l(r)
y