COMPUTER VISION
Multi-view Geometry

Emanuel Aldea <emanuel.aldea@u-psud.fr>
http://hebergement.u-psud.fr/emi/

Computer Science and Multimedia Master - University of Pavia

Context of pose estimation

Why do we need anything beside the existing algorithms ?

- Generic pose estimation and refinement algorithms fail in some contexts, e.g. :

Context of pose estimation

Why do we need anything beside the existing algorithms ?

- Generic pose estimation and refinement algorithms fail in some contexts, e.g. :
  - Large homogeneous areas (ground, facades)

Context of pose estimation

Why do we need anything beside the existing algorithms ?

- Generic pose estimation and refinement algorithms fail in some contexts, e.g. :
  - Large homogeneous areas (ground, facades)
  - Repetitive static patterns (arches, window corners etc.)

Context of pose estimation

Why do we need anything beside the existing algorithms ?

- Generic pose estimation and refinement algorithms fail in some contexts, e.g. :
  - Large homogeneous areas (ground, facades)
  - Repetitive static patterns (arches, window corners etc.)
  - Similarity of people body parts

Context of pose estimation

Why do we need anything beside the existing algorithms ?

- Generic pose estimation and refinement algorithms fail in some contexts, e.g. :
  - Large homogeneous areas (ground, facades)
  - Repetitive static patterns (arches, window corners etc.)
  - Similarity of people body parts
  - Wide baseline : perspective change, strong occlusions

Context of pose estimation

Why do we need anything beside the existing algorithms ?

- Generic pose estimation and refinement algorithms fail in some contexts, e.g. :
  - Large homogeneous areas (ground, facades)
  - Repetitive static patterns (arches, window corners etc.)
  - Similarity of people body parts
  - Wide baseline : perspective change, strong occlusions


Camera-IMU fusion for localization

Why is image based localization powerful ?

- Affordable in terms of hardware and computational cost

Camera-IMU fusion for localization

Why is image based localization powerful ?

- Affordable in terms of hardware and computational cost
- Major issue when the scene is not well textured : hard to estimate the reliability of the estimation

Camera-IMU fusion for localization

Why is image based localization powerful ?

- Affordable in terms of hardware and computational cost
- Major issue when the scene is not well textured : hard to estimate the reliability of the estimation
- Minor issue : scale must be estimated separately (i.e. the norm of the translation is unknown)

Camera-IMU fusion for localization

Why is image based localization powerful ?

- Affordable in terms of hardware and computational cost
- Major issue when the scene is not well textured : hard to estimate the reliability of the estimation
- Minor issue : scale must be estimated separately (i.e. the norm of the translation is unknown)
- Benefit of coupling with IMU and GPS : avoid faulty results

Camera-IMU fusion for localization

Why is image based localization powerful ?

- Affordable in terms of hardware and computational cost
- Major issue when the scene is not well textured : hard to estimate the reliability of the estimation
- Minor issue : scale must be estimated separately (i.e. the norm of the translation is unknown)
- Benefit of coupling with IMU and GPS : avoid faulty results

Single image based relative pose estimation

- Sensor performance : reliable but mediocre (low cost equipment)
- We know that the vision estimation is often very inaccurate

Camera-IMU fusion for localization

The skeleton of an M-Estimator approach

Identify a solution close to the sensor pose which is guided by matches from images :

$$\hat{s} = \arg\min_s \left\{ c \left(\sum_{k \in \Omega} w(k)(1 - g(k,s)) \right) + \lambda(s)^2 \right\} \tag{1}$$

Details regarding the terms :

- Ω is the set of potentially correct associations, and $w(k)$ measures the visual quality of the association k
- $g(k, s)$ evaluates the agreement between the current pose s and the association k
- $\lambda(s)$ is a measure of the proximity of the solution to the sensor pose
- c controls the relative importance of the regularisaton and data attachment terms

Camera-IMU fusion for localization

The skeleton of an M-Estimator approach

Identify a solution close to the sensor pose which is guided by matches from images :

$$\hat{s} = \arg\min_{s} \left\{ c \left(\sum_{k \in \Omega} w(k)(1 - g(k, s)) \right) + \lambda(s)^2 \right\} \tag{1}$$

Details regarding the terms :

- Ω is the set of potentially correct associations, and $w(k)$ measures the visual quality of the association k
- $g(k, s)$ evaluates the agreement between the current pose s and the association k
- $\lambda(s)$ is a measure of the proximity of the solution to the sensor pose
- c controls the relative importance of the regularisaton and data attachment terms

Initialization :

- these types of optimizations are non-convex, and thus sensitive to the initialization
- stochastic initialization by sampling poses around the prior
- aims to draw a candidate in the bassin of attraction of the estimator
- problem if the sensor information is not sufficient to build a prior

Camera-IMU fusion for localization

The agreement function $g(k, s)$

$$g(k,s) = \exp\left(-\frac{d(k,s)^2}{2\sigma_h^2}\right) \tag{2}$$

The distance $d(k, s)$ is an image space error in k when we consider s. The parameter σ_h has an important impact on the profile of the energy (the smaller it is, the more sensitive the functional).

The visual quality $w(k)$

- related to how similar p and p' are visually, based on a descriptor distance $d(p, p')$
- a robust way to define $w(k)$ in terms of the two closest distances between p and any p' :

$$w_v(k) = 1 - \frac{d_{1NN}(k)}{d_{2NN}(k)}$$

Camera-IMU fusion for localization

The agreement function $g(k, s)$

$$g(k, s) = \exp\left(-\frac{d(k, s)^2}{2\sigma_h^2}\right) \tag{2}$$

The distance $d(k, s)$ is an image space error in k when we consider s. The parameter σ_h has an important impact on the profile of the energy (the smaller it is, the more sensitive the functional).

The visual quality $w(k)$

- related to how similar p and p' are visually, based on a descriptor distance $d(p, p')$
- a robust way to define $w(k)$ in terms of the two closest distances between p and any p' :

$$w_v(k) = 1 - \frac{d_{1NN}(k)}{d_{2NN}(k)}$$

The proximity measure $\lambda(s)$

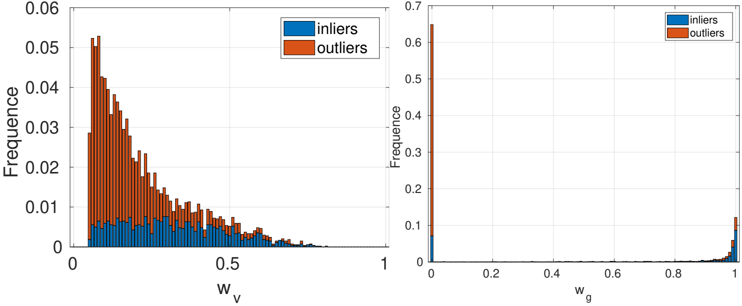
- defined as a Mahalanobis distance between s and the prior s_0 (avec $\delta s = s - s_0$) :

$$\lambda(s) = \frac{1}{|s|}\sqrt{\delta s^T \Sigma_{s_0}^{-1} \delta s}$$

Adapting the method for a specific context

Learning the weights

- The $w_v(k)$ is widely used but it exhibits known limitations in urban environments
- (Yi et al., CVPR18) proposed a neural network which estimates the correspondence weights $w_g(k)$ based on a learnt global coherence
- The two algorithms have fundamentally different behaviors :

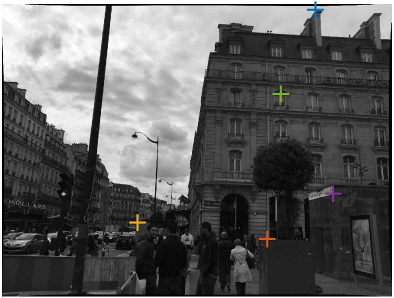


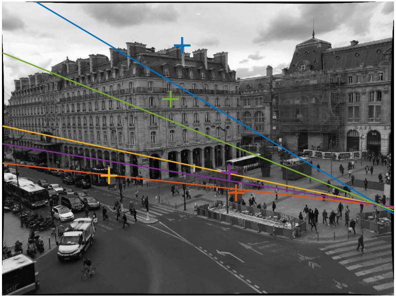
- Relying on a composite weight (stricter than the sum) improves significantly the performance of the M-Estimator

Example : static camera image

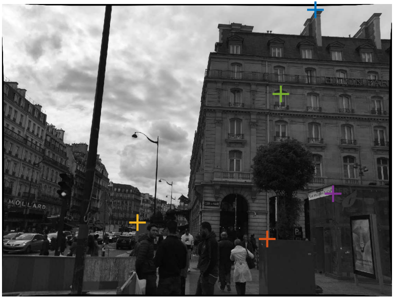
Example : dynamic camera image

Pose estimation and epipole with pure vision





Pose estimation and epipole with sensor-vision fusion



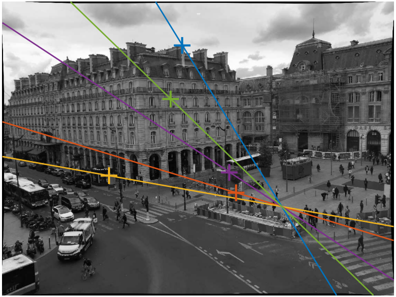
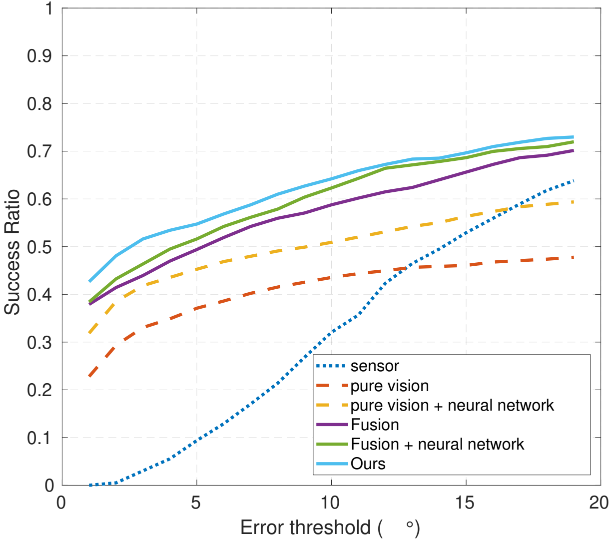
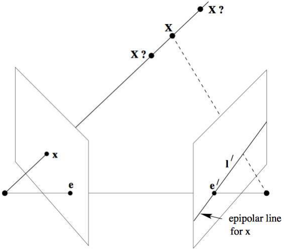


Figure with expected performance



Triangulation - the building block of 3D reprojections

We have the pose R, t' between cameras and the projection locations x, x'. What now ?



Get X : triangulate the point in 3D

Triangulation - the building block of 3D reprojections

We have the pose R, t' between cameras and the projection locations x, x'. What now ?

Get X : triangulate the point in 3D

- Back to our stereo projection equations :

$$\lambda x = KX \quad \lambda' x' = K'(RX + t)$$

Triangulation - the building block of 3D reprojections

We have the pose R, t' between cameras and the projection locations x, x'. What now ?

Get X : triangulate the point in 3D

- Back to our stereo projection equations :

$$\lambda x = KX \quad \lambda' x' = K'(RX + t)$$

- We have five scalar unknowns and six equations - a direct approach is possible by solving an overdetermined linear system

Triangulation - the building block of 3D reprojections

We have the pose R, t' between cameras and the projection locations x, x'. What now ?

Get X : triangulate the point in 3D

- Back to our stereo projection equations :

$$\lambda \mathrm{x} = \mathrm{K}\mathrm{X} \quad \lambda' \mathrm{x}' = \mathrm{K}'(\mathrm{R}\mathrm{X} + \mathrm{t})$$

- We have five scalar unknowns and six equations - a direct approach is possible by solving an overdetermined linear system
- There are other algorithms which are more accurate, but costlier
  Hartley, R. I., Sturm, P. (1997). Triangulation. Computer vision and image understanding, 68(2), 146-157

  Lindstrom, Peter. "Triangulation made easy." In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, pp. 1554-1561

Triangulation - the building block of 3D reprojections

We have the pose R, t' between cameras and the projection locations x, x'. What now ?

Get X : triangulate the point in 3D

- Back to our stereo projection equations :

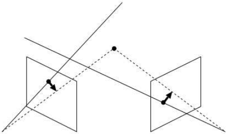
$$\lambda x = KX \quad \lambda' x' = K'(RX + t)$$

- We have five scalar unknowns and six equations - a direct approach is possible by solving an overdetermined linear system
- There are other algorithms which are more accurate, but costlier
  Hartley, R. I., Sturm, P. (1997). Triangulation. Computer vision and image understanding, 68(2), 146-157
  Lindstrom, Peter. "Triangulation made easy." In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, pp. 1554-1561
- The linear approach is reasonably good, and it is effective especially if used as an initialization for a nonlinear refinement (as we will see in the following slides)

Triangulation - how to use multiple views

If we have multiple views, the unknown X_j may be constrained by multiple observations $\mathrm{z}_{j,\tau}$ from cameras C_τ characterized by some pose parametrization s_τ . How to use them effectively together ?

Nonlinear optimization

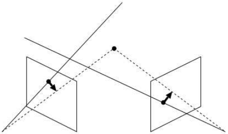


Triangulation - how to use multiple views

If we have multiple views, the unknown X_j may be constrained by multiple observations $z_{j,\tau}$ from cameras C_τ characterized by some pose parametrization s_τ . How to use them effectively together ?

Nonlinear optimization

- Analytical solutions are not practical, in most cases we solve the optimization iteratively



Triangulation - how to use multiple views

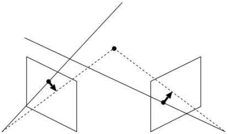
If we have multiple views, the unknown X_j may be constrained by multiple observations $\mathrm{z}_{j,\tau}$ from cameras C_τ characterized by some pose parametrization s_τ . How to use them effectively together ?

Nonlinear optimization

- Analytical solutions are not practical, in most cases we solve the optimization iteratively
- We define an error related to each of the observation, i.e. the distance between the observation and the projection of X_j : $e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_j) = \mathrm{z}_j - g(\mathrm{s}_\tau, \mathrm{X}_j)$, where g is the camera projection function. Then, we have :

$$\hat{\mathrm{X}}_j = \arg\min_{\mathrm{X}_j} \sum_\tau e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_j)^T e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_j)$$

- Use Gauss-Newton or LM (usually the optimum is not far from a reasonable initialization)



Triangulation - how to use multiple views

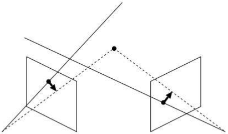
If we have multiple views, the unknown X_j may be constrained by multiple observations $z_{j,\tau}$ from cameras C_τ characterized by some pose parametrization s_τ . How to use them effectively together ?

Nonlinear optimization

- Analytical solutions are not practical, in most cases we solve the optimization iteratively
- We define an error related to each of the observation, i.e. the distance between the observation and the projection of X_j : $e(s_\tau, X_j, z_j) = z_j - g(s_\tau, X_j)$, where g is the camera projection function. Then, we have :

$$\hat{X}_j = \arg\min_{X_j} \sum_{\tau} e(s_\tau, X_j, z_j)^T e(s_\tau, X_j, z_j)$$

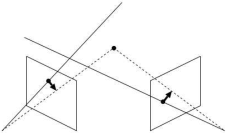
- Use Gauss-Newton or LM (usually the optimum is not far from a reasonable initialization)
- More than one 3D point may be refined, but in this way the optimizations are decoupled



Pose estimation - how to use multiple views

Opposite problem : we have a set of 3D points $\mathrm{X_j}$ (computed previously) which are visible from camera C_τ. Based on current observations $\mathrm{z}_{j,\tau}$ from C_τ we would like to estimate its pose s_τ .

Nonlinear optimization



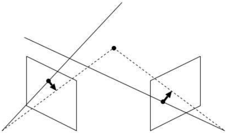
Pose estimation - how to use multiple views

Opposite problem : we have a set of 3D points X_j (computed previously) which are visible from camera C_τ. Based on current observations $\mathsf{z}_{j,\tau}$ from C_τ we would like to estimate its pose s_τ .

Nonlinear optimization

- We define an error related to each of the observations, i.e. the distance between the observation and the projection of X_j : $e(\mathsf{s}_\tau, \mathsf{X}_j, \mathsf{z}_{j,\tau}) = \mathsf{z}_{j,\tau} - g(\mathsf{s}_\tau, \mathsf{X}_j)$, where g is the camera projection function. Then, we have :

$$\hat{\mathsf{s}}_\tau = \arg\min_{\mathsf{s}_\tau} \sum_j e(\mathsf{s}_\tau, \mathsf{X}_j, \mathsf{z}_{j,\tau})^T e(\mathsf{s}_\tau, \mathsf{X}_j, \mathsf{z}_{j,\tau})$$



Pose estimation - how to use multiple views

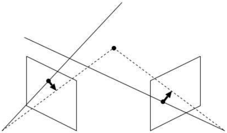
Opposite problem : we have a set of 3D points $\mathrm{X_j}$ (computed previously) which are visible from camera C_τ. Based on current observations $\mathrm{z}_{j,\tau}$ from C_τ we would like to estimate its pose s_τ .

Nonlinear optimization

- We define an error related to each of the observations, i.e. the distance between the observation and the projection of X_j : $e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_{j,\tau}) = \mathrm{z}_{j,\tau} - g(\mathrm{s}_\tau, \mathrm{X}_j)$, where g is the camera projection function. Then, we have :

$$\hat{\mathrm{s}}_\tau = \arg\min_{\mathrm{s}_\tau} \sum_j e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_{j,\tau})^T e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_{j,\tau})$$

- Use Gauss-Newton or LM, but the initialization is very important. Two strategies help :



Pose estimation - how to use multiple views

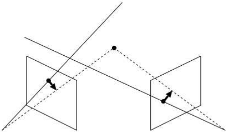
Opposite problem : we have a set of 3D points $\mathrm{X_j}$ (computed previously) which are visible from camera C_τ. Based on current observations $\mathrm{z}_{j,\tau}$ from C_τ we would like to estimate its pose s_τ .

Nonlinear optimization

- We define an error related to each of the observations, i.e. the distance between the observation and the projection of X_j : $e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_{j,\tau}) = \mathrm{z}_{j,\tau} - g(\mathrm{s}_\tau, \mathrm{X}_j)$, where g is the camera projection function. Then, we have :

$$\hat{\mathrm{s}}_\tau = \arg\min_{\mathrm{s}_\tau} \sum_j e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_{j,\tau})^T e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_{j,\tau})$$

- Use Gauss-Newton or LM, but the initialization is very important. Two strategies help :
  - if the camera is moving, predict the current location based on its previous trajectory



Pose estimation - how to use multiple views

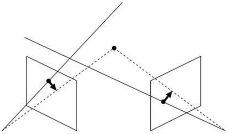
Opposite problem : we have a set of 3D points $\mathrm{X_j}$ (computed previously) which are visible from camera C_τ. Based on current observations $\mathrm{z}_{j,\tau}$ from C_τ we would like to estimate its pose s_τ .

Nonlinear optimization

- We define an error related to each of the observations, i.e. the distance between the observation and the projection of X_j : $e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_{j,\tau}) = \mathrm{z}_{j,\tau} - g(\mathrm{s}_\tau, \mathrm{X}_j)$, where g is the camera projection function. Then, we have :

$$\hat{\mathrm{s}}_\tau = \arg\min_{\mathrm{s}_\tau} \sum_j e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_{j,\tau})^T e(\mathrm{s}_\tau, \mathrm{X}_j, \mathrm{z}_{j,\tau})$$

- Use Gauss-Newton or LM, but the initialization is very important. Two strategies help :
  - if the camera is moving, predict the current location based on its previous trajectory
  - from the projection of three 3D points in space and their projections, one may compute the camera pose in a closed form (the P3P problem)



Limitations of previous approaches

Assumptions :

Limitations of previous approaches

Assumptions :

- for triangulation : we assume that the pose is correctly estimated

Limitations of previous approaches

Assumptions :

- for triangulation : we assume that the pose is correctly estimated
- for pose estimation : we assume that the 3D locations are accurate

Limitations of previous approaches

Assumptions :

- for triangulation : we assume that the pose is correctly estimated
- for pose estimation : we assume that the 3D locations are accurate
- in reality all estimations we perform are noisy

Limitations of previous approaches

Assumptions :

- for triangulation : we assume that the pose is correctly estimated
- for pose estimation : we assume that the 3D locations are accurate
- in reality all estimations we perform are noisy
- if we also apply the process iteratively (triangulation, pose estimation and repeat) the errors will be amplified (drift)

Global optimization - initial step

Since computational power is widely available for autonomous systems, we favour a solution which minimizes jointly with respect to the point locations and to the poses.

Initial step :

Global optimization - initial step

Since computational power is widely available for autonomous systems, we favour a solution which minimizes jointly with respect to the point locations and to the poses.

Initial step :

- we will just add a new unknown pose to the previous set of variables and refine it :

$$\hat{s}_\tau = \arg\min_{s_\tau} \sum_j e(s_\tau, X_j, z_{j,\tau})^T e(s_\tau, X_j, z_{j,\tau})$$

Global optimization - initial step

Since computational power is widely available for autonomous systems, we favour a solution which minimizes jointly with respect to the point locations and to the poses.

Initial step :

- we will just add a new unknown pose to the previous set of variables and refine it :

$$\hat{s}_\tau = \arg\min_{s_\tau} \sum_j e(s_\tau, X_j, z_{j,\tau})^T e(s_\tau, X_j, z_{j,\tau})$$

- observation : this step does not modify X

Global optimization - initial step

Since computational power is widely available for autonomous systems, we favour a solution which minimizes jointly with respect to the point locations and to the poses.

Initial step :

- we will just add a new unknown pose to the previous set of variables and refine it :
$$\hat{s}_\tau = \arg\min_{s_\tau} \sum_j e(s_\tau, X_j, z_{j,\tau})^T e(s_\tau, X_j, z_{j,\tau})$$
- observation : this step does not modify X
- the interest of the initial step is just to provide a quality initialization for s_τ as $\hat{s}_t$

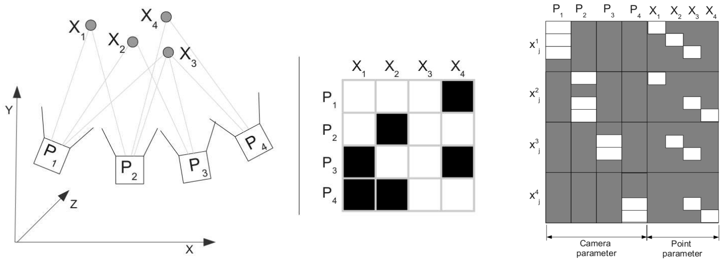
Global optimization - final step

We compute the MAP (Maximum A Posteriori) for the maximum amount of preliminary estimations and observations that we have at that moment (brutal, massive optimization). The solution we search this time is provided by :

$$\tilde{\mathsf{S}}_{0:t}, \tilde{\mathsf{X}} = \arg\min_{\mathsf{S}_{0:t}, \mathsf{X}} \sum_{\tau=0}^{T} \sum_{j=1}^{M} e(\mathsf{s}_\tau, \mathsf{X}_{j,\tau}, z_{j,\tau})^T \; e(\mathsf{s}_\tau, \mathsf{X}_{j,\tau}, z_{j,\tau})$$

The complexity of this algorithm, once we exploit the sparseness of its Jacobian : $O(T^3 + MT^2)$, which is very interesting since $M \gg T$.

Towards real time reconstruction



An example of configuration : 5207 3D points, 54 poses, 24609 projections, 15945 variables, 21 it., 7.99 sec.
Not fast enough !

- Selection of key-frames
- Parallel execution of tracking et BA (initial and final steps)
- Limit the number of iterations (when needed)
- Local Bundle Adjustment

Typical architecture for RT optimization