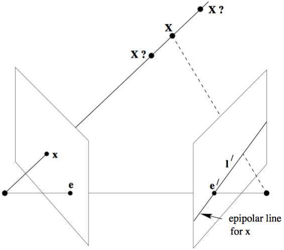
COMPUTER VISION
Multi-view Geometry

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

Computer Science and Multimedia Master - University of Pavia

Triangulation - the building block of 3D reprojections

We have the pose $\mathbf{R}, \mathbf{t}'$ between cameras and the projection locations $\mathbf{x}, \mathbf{x}'$. What now ?



Get $\mathbf{X}$: triangulate the point in 3D

Triangulation - the building block of 3D reprojections

We have the pose $\mathbf{R}, \mathbf{t}'$ between cameras and the projection locations $\mathbf{x}, \mathbf{x}'$. What now ?

Get $\mathbf{X}$: triangulate the point in 3D

- Back to our stereo projection equations :

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

Triangulation - the building block of 3D reprojections

We have the pose $\mathbf{R}, \mathbf{t}'$ between cameras and the projection locations $\mathbf{x}, \mathbf{x}'$. What now ?

Get $\mathbf{X}$: triangulate the point in 3D

- Back to our stereo projection equations :

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

- 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 $\mathbf{R}, \mathbf{t}'$ between cameras and the projection locations $\mathbf{x}, \mathbf{x}'$. What now ?

Get $\mathbf{X}$: triangulate the point in 3D

- Back to our stereo projection equations :

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

- 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 $\mathbf{R}, \mathbf{t}'$ between cameras and the projection locations $\mathbf{x}, \mathbf{x}'$. What now ?

Get $\mathbf{X}$: triangulate the point in 3D

- Back to our stereo projection equations :

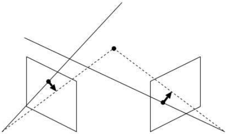
$$\lambda \mathbf{x} = \mathbf{K}\mathbf{X} \quad \lambda' \mathbf{x}' = \mathbf{K}'(\mathbf{R}\mathbf{X} + \mathbf{t})$$

- 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 $\mathbf{X}_j$ may be constrained by multiple observations $\mathbf{z}_{j,\tau}$ from cameras C_τ characterized by some pose parametrization $\mathbf{s}_\tau$. How to use them effectively together ?

Nonlinear optimization

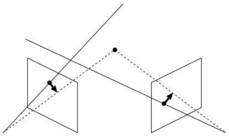


Triangulation - how to use multiple views

If we have multiple views, the unknown $\mathbf{X}_j$ may be constrained by multiple observations $\mathbf{z}_{j,\tau}$ from cameras C_τ characterized by some pose parametrization $\mathbf{s}_\tau$. 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

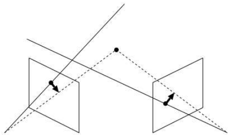
If we have multiple views, the unknown $\mathbf{X}_j$ may be constrained by multiple observations $\mathbf{z}_{j,\tau}$ from cameras C_τ characterized by some pose parametrization $\mathbf{s}_\tau$. How to use them effectively together ?

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 $\mathbf{X}_j$: $e(\mathbf{s}_\tau, \mathbf{X}_j, \mathbf{z}_j) = \mathbf{z}_j - g(\mathbf{s}_\tau, \mathbf{X}_j)$, where g is the camera projection function. Then, we have :

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

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



Triangulation - how to use multiple views

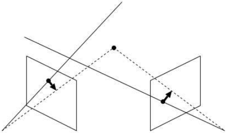
If we have multiple views, the unknown $\mathbf{X}_j$ may be constrained by multiple observations $\mathbf{z}_{j,\tau}$ from cameras C_τ characterized by some pose parametrization $\mathbf{s}_\tau$. How to use them effectively together ?

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 $\mathbf{X}_j$: $e(\mathbf{s}_\tau, \mathbf{X}_j, \mathbf{z}_j) = \mathbf{z}_j - g(\mathbf{s}_\tau, \mathbf{X}_j)$, where g is the camera projection function. Then, we have :

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

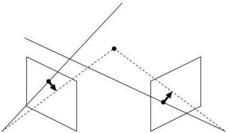
- Use Gauss-Newton or LM (usually the optimum is not far from a reasonable initialization)
- More than one 3D point may be refined, but in this way the optimizations are decoupled



Pose estimation - how to use multiple views

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

Nonlinear optimization



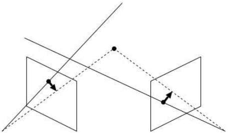
Pose estimation - how to use multiple views

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

Nonlinear optimization

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

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



Pose estimation - how to use multiple views

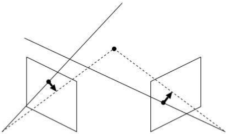
Opposite problem : we have a set of 3D points $\mathbf{X_j}$ (computed previously)which are visible from camera C_τ. Based on current observations $\mathbf{z}_{j,\tau}$ from C_τ we would like to estimate its pose $\mathbf{s}_\tau$.

Nonlinear optimization

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

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

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



Pose estimation - how to use multiple views

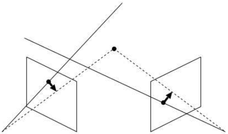
Opposite problem : we have a set of 3D points $\mathbf{X_j}$ (computed previously)which are visible from camera C_τ. Based on current observations $\mathbf{z}_{j,\tau}$ from C_τ we would like to estimate its pose $\mathbf{s}_\tau$.

Nonlinear optimization

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

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

- 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

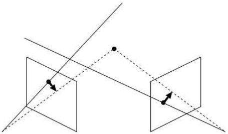
Opposite problem : we have a set of 3D points $\mathbf{X_j}$ (computed previously)which are visible from camera C_τ. Based on current observations $\mathbf{z}_{j,\tau}$ from C_τ we would like to estimate its pose $\mathbf{s}_\tau$.

Nonlinear optimization

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

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

- 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 :

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

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 :

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

- observation : this step does not modify $\mathbf{X}$

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 :

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

- observation : this step does not modify $\mathbf{X}$
- the interest of the initial step is just to provide a quality initialization for $\mathbf{s}_\tau$ as $\hat{\mathbf{s}}_t$

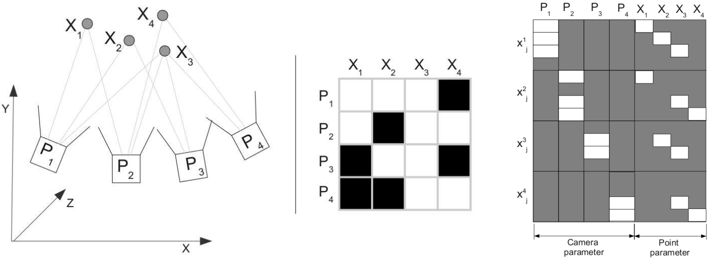
Global optimization - final step

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 :

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

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