Bundle Adjustment - A modern Synthesis

Ariyan Zarei

University of Arizona

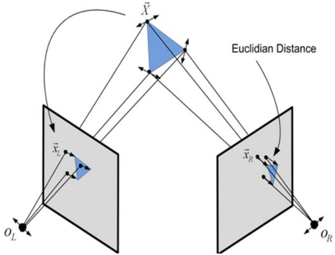
ariyanzarei@email.arizona.edu

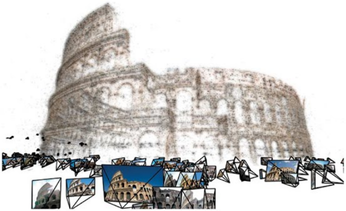
November 25, 2020

Overview

Bundle Adjustment Problem Formulation

Visual Reconstruction and Bundle Adjustment





Definitions

- **Visual Reconstruction:** Recover a model of a 3D scene from multiple images.
- **Scene Model:** Collection of isolated 3D features, e.g., points, lines, etc.
- **Bundle Adjustment:** Problem of refining a visual reconstruction model to produce jointly optimal 3D structure and viewing parameter (camera pose/calibration) estimates.
- **jointly**: Solution is simultaneously optimal with respect to both structure and camera variations.
- **optimal**: Parameter estimates are found by minimizing some cost function that quantifies the model fitting error.
- **Bundle** in the name refers to the bundles of light rays leaving each 3D feature and converging on each camera center.

Why use Bundle Adjustment

Reasons for using Bundle Adjustment over alternative adjustment methods

- Flexibility: handles very wide variety of sources of information (like priors).
- Accuracy: provide precise and easy to interpret results.
- Efficiency: Use rapidly convergent numerical methods and make good use of sparsity.

Notations (abstract)

- State Vector X: large vector of all parameters
  - 3D locations of the keypoints
  - Camera parameters
- $\underline{Z}$: observations, measured image features.
- $Z = Z(X)$: predicted values corresponding to the observation from parameters X.
- $\Delta Z(X) = \underline{Z} - Z(X)$: residual prediction error.
- $f(X) = f(predz(X))$: cost function.
- $g = \frac{df}{dX}$: Gradient of the cost function.
- $H = \frac{d^2 f}{dX^2}$: Hessian of the cost function.
- $J = \frac{dZ}{dX}$: Jacobian of the observation w.r.t. states.

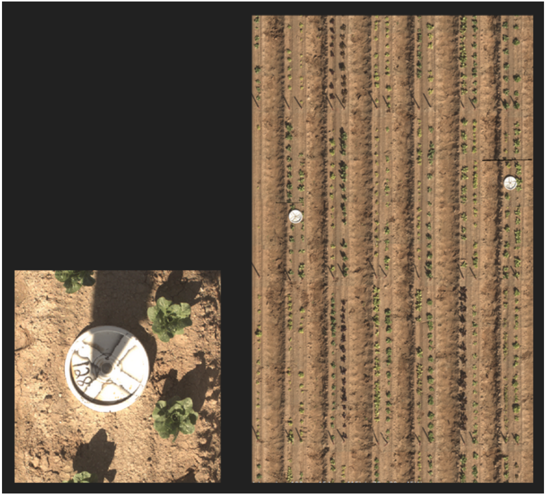
The Projection Model

- Camera Models
  - Perspective Projection (Homography)
  - Affine
  - Orthographic Projection
  - Pushbroom Models (for Pushbroom scanners)
- Scene Model
  - $\forall p \in \{1, \ldots, n\}$ X_p: Individual static 3D features. n is the total number of features in the scene.
  - $\forall i \in \{1, \ldots, m\}$ P_i: Camera pose and internal camera calibration parameters for image i. m is the total number of images taken.
  - $\forall c \in \{1, \ldots, k\}$ C_c: Other calibration parameters.
  - $\underline{x}_{ip}$: Uncertain measurement of 3D feature X_p in image i.
  - $x_{ip} = x(C_c, P_i, X_p)$: x_{ip} is the true image of feature X_p in image i which can be constructed using a predictive model of the parameters.

The Projection Model

- Feature Prediction Error: $\underline{x}_{ip} - x_{ip} = \underline{x}_{ip} - x(C_c, P_i, X_p)$
- Other measurements in the 3D world can be included.
- Estimate unknown 3D features and camera parameters by minimizing some measure of total prediction error.
- Bundle Adjustment
  - Optimizing Nonlinear cost function
  - Over Nonlinear space of parameters

Prior knowledge



Cost function and Error Modeling

- Cost function
  - Quantifies total prediction error (reprojection error)
- Allow for presence of outliers
  - Robust statistically-based error metrics
  - Based on total (inlier+outlier) log likelihood
- Parameter estimation
  - optimal point estimator for cost function minimization
  - Maximum likelihood (ML)
  - Maximum a posteriori (MAP)
    - Include priors

Parameter Estimation Methods

- Nonlinear Least Squares
- Robustified Least Squares
- Intensity-based Methods
- Implicit Models

Nonlinear Least Squares

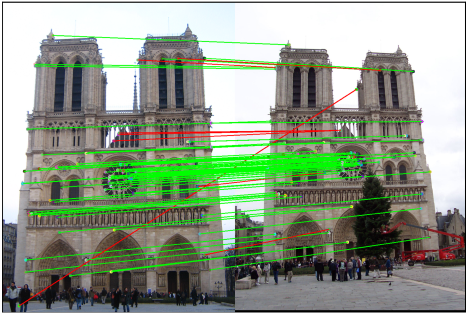
$$f(x) = \frac{1}{2}\sum_i \Delta Z_i(X)^\top W_i \Delta Z_i(X)$$

$$\Delta Z_i(X) = \underline{Z}_i - Z_i(X)$$

- $Z_i = Z_i(X)$: predicted value of the feature i corresponding to the observation from parameters X.
- $\underline{Z}_i$: observed value of feature i.
- W_i: symmetric positive definite weights.
- Minimizes the weighted sum of squared error.
- Equivalent: observations perturbed by Gaussian noise with mean zero and covariance W_i^{-1}.

Robustified Least Squares

- Nonlinear Least Squares are highly sensitive to outliers. We want to damper the effects of outliers.
- Example of outliers: incorrect SIFT matchings
- RANSAC
- observations are considered independent (independence groups)



Robustified Least Squares

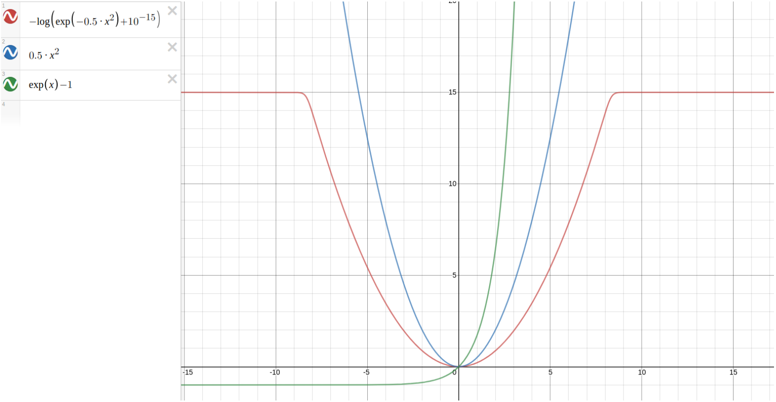
- cost functions can have any form. One natural family of cost functions are **radial distributions**:

$$f_i(x) = \frac{1}{2}\rho_i(\Delta Z_i(X)^\top W_i \Delta Z_i(X)^2)$$

- ρ_i can be any increasing function with $\rho_i(0) = 0$ and $\frac{d}{ds}\rho_i(0) = 1$.
- Model the error distribution as a Gaussian central peak plus a uniform background of outliers.

$$f(x) = -\sum_i log(e^{-\frac{1}{2}(\Delta Z_i(X)^\top W_i \Delta Z_i(X))^2} + \epsilon)$$

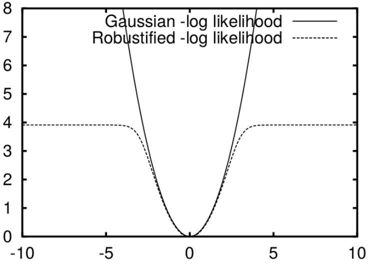
Robustified Least Squares



Robustified Least Squares

$$f(x) = -\sum_i \sum_p log(e^{-\frac{1}{2}(\Delta x_{ip}^\top W_{ip} \Delta x_{ip})^2} + \epsilon)$$

- $\Delta x_{ip} = \underline{x}_{ip} - x_{ip}$: difference of the observed and predicted feature locations.
- ϵ parametrizes the frequency of outliers.



Other Methods

- Intensity-based Models: observables are intensities (of the patches of images)
- Implicit Models: Relations between predictions and parameters are not explicit.
  - We used to have $Z = Z(X)$ in which Z is expressed explicitly in terms of the parameters.
  - Here we have $h(X, Z) = 0$.
  - two approaches to handle implicit models
    - Nuisance parameters: convert to a constrained optimization by adding Z as Nuisance parameters. Minimize $f(\underline{Z} - Z)$ over (X, Z) subject to $h(X, Z) = 0$.
    - Reduction: Consider $h(X, \underline{Z})$ instead of $\underline{Z}$

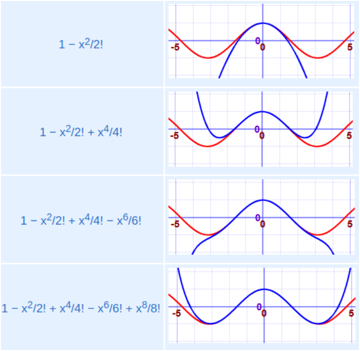
Numerical Optimization

- Minimize cost function $f(X)$ starting from some approximations for X.
- Not possible to minimize in closed form.
- Take steps in the directions of reducing cost function locally and possibly reducing it globally.
- Take advantage of Taylor series.

Second Order Methods

- Taylor Series

$$f(x) \approx f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \dots$$



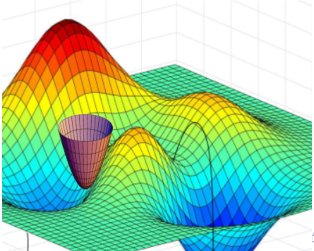
Second Order Methods

- Quadratic Taylor Approximation of the cost function

$$f(X + \delta X) \approx f(X) + g^\top \delta X + \frac{1}{2}\delta X^\top H \delta X$$

- This is a local model with a global optimum.
- We proceed by taking the Newton's step:

$$\delta X = -H^{-1} g$$



Second Order Methods

Remarks:

- Needs Step Control
- Computation Complexity of each step is $O(n^3)$.
- Calculating H is not easy.
- Starting point is very important. Convergence becomes difficult when starting point is far from global minimum.

Step Control

- Newton's method might fail
  - Converge to saddle point
  - Occasionally true cost might not decrease when approximate cost decreases
- Descent Direction
  - A direction with a non-negligible component down the local cost gradient.
  - Take a combination of Newton's step and gradient direction

$$(H + \lambda W)\delta X = -g$$

  - λ controls the combination and W is a PSD weight matrix.
  - Trust Region Methods: Choose maximum size of the step dynamically
  - Levenberg-Marquardt Methods: λ increases and decreases based on the whether the approximation gets better or worse.

Gauss-Newton

- Jacobian of the predictions w.r.t. the parameters

$$J = \frac{dZ}{dX} = \begin{bmatrix} \frac{dZ_1}{dX_1} & \frac{dZ_1}{dX_2} & \cdots & \frac{dZ_1}{dX_n} \\ \frac{dZ_2}{dX_1} & \frac{dZ_2}{dX_2} & \cdots & \frac{dZ_2}{dX_n} \\ \vdots & \vdots & & \vdots \\ \frac{dZ_m}{dX_1} & \frac{dZ_m}{dX_2} & \cdots & \frac{dZ_m}{dX_n} \end{bmatrix}$$

- Gradient and Hessian can be written in terms of Jacobian

$$g = \frac{df}{dX} = \Delta Z^\top W J$$

$$H = \frac{d^2 f}{dX^2} = J^\top W J + \sum_i \left(\Delta Z^\top W\right)_i \frac{d^2 Z_i}{dX^2}$$

Gauss-Newton

$$g = \frac{df}{dX} = \Delta Z^\top W J$$

$$H = \frac{d^2 f}{dX^2} = J^\top W J + \sum_i \left(\Delta Z^\top W\right)_i \frac{d^2 Z_i}{dX^2}$$

- We can drop $\sum_i \left(\Delta Z^\top W\right)_i \frac{d^2 Z_i}{dX^2}$ if either
  - Prediction Error ΔZ is small
  - Model is nearly linear or $\frac{d^2 Z_i}{dX^2} \approx 0$
- Results would be Gauss-Newton approximation

$$\left(J^\top W J\right) \delta X = -J^\top W \Delta Z$$

Gauss-Newton

- Gauss-Newton is an approximation of the SSE

$$\left(J^\top WJ\right)\delta X = -J^\top W\Delta Z$$

- Very common specially in [bundle] adjustment methods.
- Disadvantages
  - Sometimes discarded terms are not negligible
  - Very sensitive to saddle point

Robustified Gauss-Newton

- Recall Least Squares and Robustified Least Squares

$$f(X) = \frac{1}{2} \sum_i \Delta Z_i(X)^\top W_i \Delta Z_i(X)$$

$$f_i(X) = \frac{1}{2} \rho_i(\Delta Z_i(X)^\top W_i \Delta Z_i(X)^2)$$

- In robustified Least Squares we introduce nonlinearity at the level of individual features.
- Disadvantages of Gauss-Newton does not extend to the robust cost functions

$$g_i = \rho_i' J_i^\top W_i \Delta Z_i$$

$$H_i = J_i^\top (\rho_i' W_i + 2\rho'' (W_i \Delta Z_i)(W_i \Delta Z_i)^\top) J_i$$

Constrained Problems

- Minimize the cost function $f(X)$ w.r.t. a set of constraints
- Two approaches
  - Lagrange Multipliers: instead of minimizing $f(X)$, minimize $f(X) + \lambda c(X)$
  - Reduce the Problem: Eliminate some of the variables by substitution and optimize over the rest.

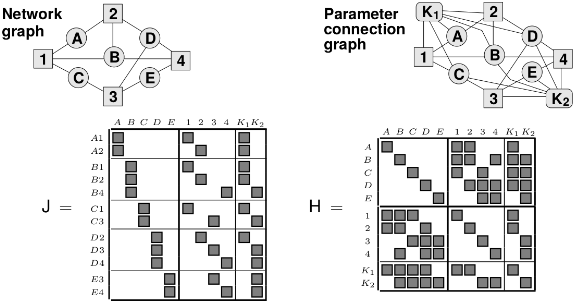
Implementation Tricks

- Exploit the structure of the problem: Use Similarity/Affine whenever possible
- Use Factorization of the matrices: Instead of computing inverse of matrices, use matrix decomposition techniques
- Use local parametrization
- Use variable scaling (like in Newton-like methods)

Implementation Tricks

- Exploit the structure of the problem: Use Similarity/Affine whenever possible
- Use Factorization of the matrices: Instead of computing inverse of matrices, use matrix decomposition techniques
- Use local parametrization
- Use variable scaling (like in Newton-like methods)

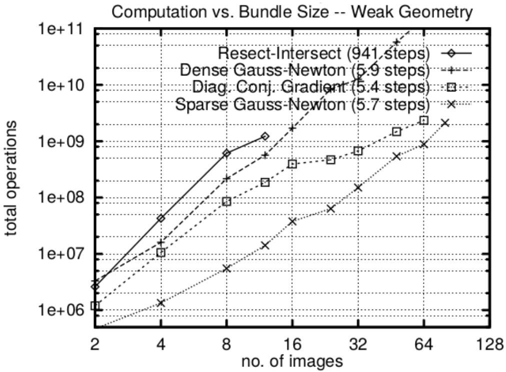
Network Structure



- Structure of the problem
- Sparsity
- Relation between parameters

Experimental and Implementation Remarks

- Take advantage of Matrix Decomposition techniques to speed up computation
- Take advantage of the sparsity of the problem
- Improve the first order methods (gradient descent) by applying multiple modifications



References

- Triggs, Bill, et al. "Bundle adjustmenta modern synthesis." International workshop on vision algorithms. Springer, Berlin, Heidelberg, 1999.

Thank you for your attention

I will post the slides to my homepage at
`http://vision.cs.arizona.edu/ariyanzarei/`