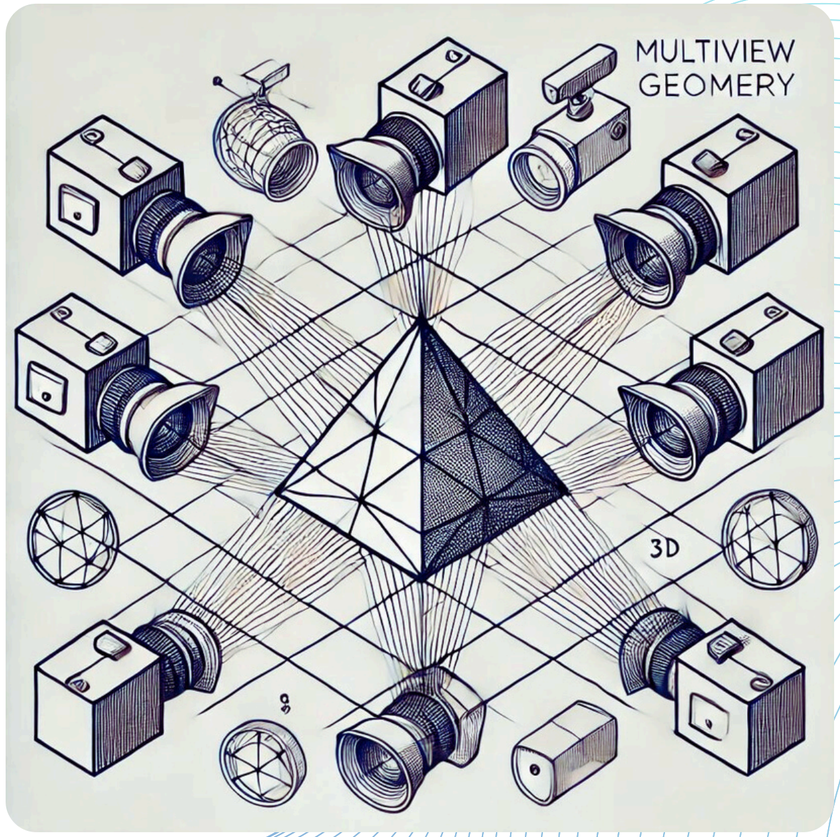
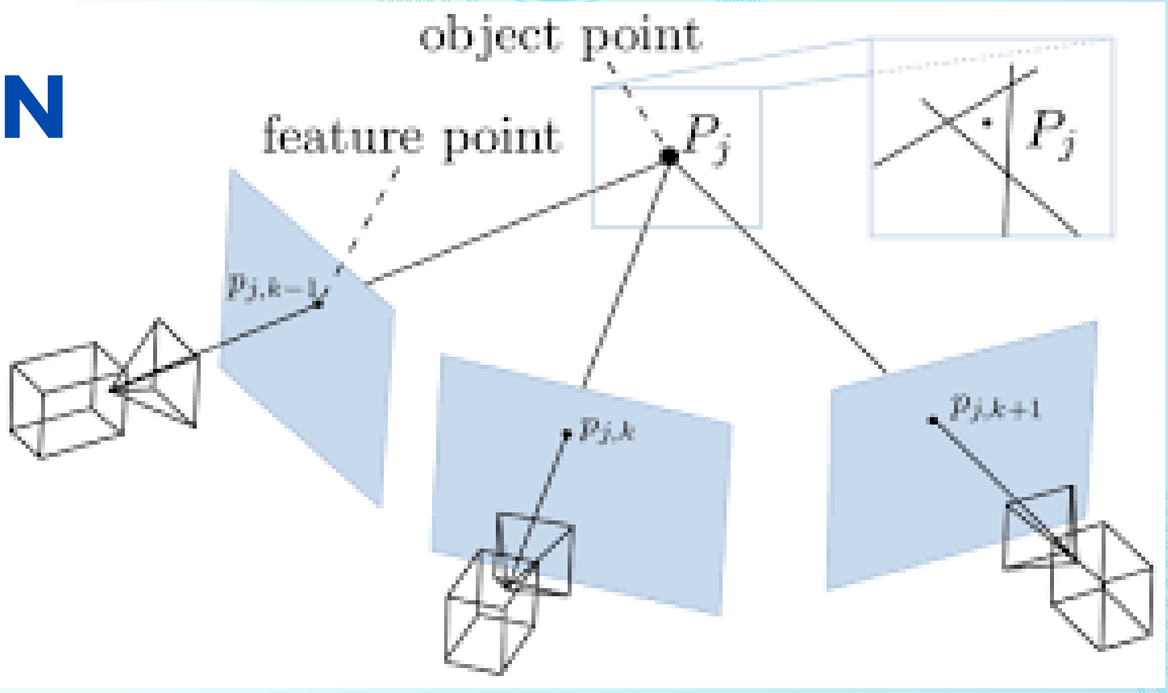
PRACTICAL GLOBAL OPTIMIZATION FOR MULTIVIEW GEOMETRY



MULTIVIEW TRIANGULATION

MULTIVIEW TRIANGULATION

Finding the best 3D position of a point from multiple camera views.



What is mean by finding best 3D position?

The "best 3D position" is the point in 3D space that best matches what all the cameras see.

1. Determining the best 3D position of a point from multiple camera views involves reconstructing the point's exact location in 3D space based on its 2D projections in different camera images.

2. Each camera captures a 2D image of the scene, and the point appears at a specific position in each image depending on the camera's angle and distance.

3. By analyzing the 2D coordinates of the point in these images, it is possible to compute the point's real-world coordinates.

PROJECTIVE GEOMETRY

Projective geometry is a method use in computer vision, enabling techniques for recovering the 3D structure of a scene from multiple images.

The core challenge in these methods involves solving complex optimization problems, which are typically non-linear and difficult to solve globally.

CAN WE USE PROJECTIVE GEOMETRY FOR TRANGULATION ???

Problem: Triangulation often gets trapped in local minima due to non-linear, non-convex nature of cost function optimization problems.

Proposed Method: Practical global optimization method

PRACTICAL GLOBAL OPTIMIZATION

Practical global optimization refers to a method that finds the best possible solution (global optimum) to certain complex problems in multiview geometry such as triangulation.

This is a scalable global optimization algorithm using branch and bound with fractional programming and convex programming.

FRACTIONAL PROGRAMMING

The goal of fractional programming is to minimize the sum of several fractions, where each fraction has a numerator function and a denominator function.

$$\min_{x} \quad \sum_{i=1}^{p} \frac{f_i(x)}{g_i(x)} \quad \text{subject to} \quad x \in D$$

To simplify the problem, introduce auxiliary variables t and s and reformulate the optimization problem in a different way. This new formulation is easier to handle because it deals with a convex set (which is easier to optimize over), and it ensures that the new problem is equivalent to the original one.

$$\begin{aligned} \min_{x,t,s} \quad & \sum_{i=1}^{p} \frac{t_i}{s_i} \\ \text{subject to} \quad & f_i(x) \le t_i \qquad g_i(x) \ge s_i \\ & x \in D \qquad (t,s) \in Q_0. \end{aligned}$$

If we solve this equation, we automatically solve original equation.

BOUNDING

Goal - provide a lower bound on the minimum value of the objective function in a given domain

Convex Envelope-

Creating a simpler, convex approximation of the original function f(x)

- **Underestimation**: The convex envelope is always less than or equal to the original function over the domain. This ensures that the convex envelope provides a valid lower bound.
- **Tightness**: The convex envelope is the tightest convex function. This means it stays as close as possible to the original function, making it a good approximation.

BRANCHING

Goal - systematically divide the search space into smaller regions (or rectangles) to narrow down the areas that may contain the global optimum

- Challenge- deciding which dimension to split
- General strategy- split along the dimensions corresponding to the denominators si of the fractional terms ti / si
- This method is effective for problems with a small number of fractions
- But it becomes impractical when dealing with many fractions

α-bisection Algorithm best solution x*

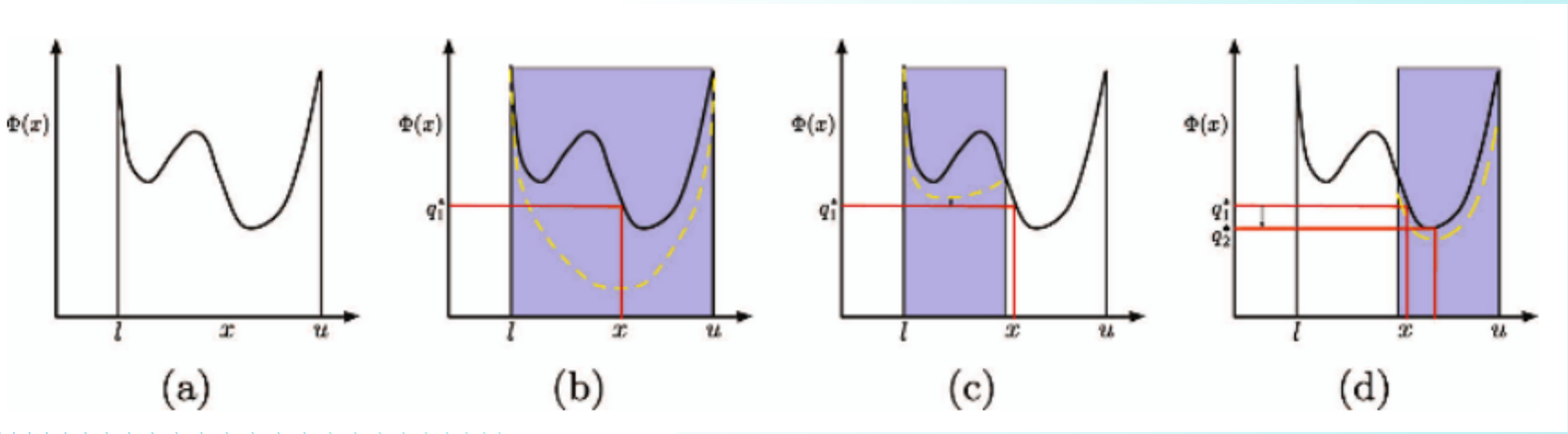
If $(x^* - L)/(U - L) < \alpha$, set the new upper bound $U_{\text{new}} = L + \alpha(U - L)$.

If $(U - x^*)/(U - L) < \alpha$, set the new lower bound $L_{\text{new}} = U - \alpha(U - L)$.

BRANCH AND BOUND THEORY

Process:

- The algorithm starts by calculating the lower bound for the entire search space.
- If the difference between the function value at the best point found so far and the lower bound is small enough, the algorithm stops.
- If not, the space is divided into smaller regions, and the process repeats.
- Any subregion that cannot contain the global minimum is discarded.



TRIANGULATION

Recovering the 3D coordinates of a point using 2D images across multiple views and the camera matrices.

Reprojection residual - difference between the observed 2D image point and the projection of the 3D point back onto the image plane using the camera matrix

$$r = \left(\frac{u - p_1 X}{p_3 X}, \frac{v - p_2 X}{p_3 X} \right)$$

(u, v) measured 2D image points

$\left(\frac{p_1 X}{p_3 X}, \frac{p_2 X}{p_3 X} \right)$ reprojected points

OBJECTIVE FUNCTION

Goal - minimize the reprojection error across multiple views N

$$\sum_{i=1}^{N} \|r_i\|_q^p$$

$\|r\|_q^p$ different norms for measuring the reprojection error

MAXIMUM LIKELIHOOD ESTIMATION

- Objective function to be minimized is derived from the probability distribution of errors (noise) in the observations.
- The choice of the noise model (Gaussian, Laplacian) defines the cost function that is minimized in the multiview geometry problem.

Gaussian Noise Model (L2, L2 Formulation)

$$p(x|x^*) = (2\pi\sigma^2)^{-1} \exp\left(-\frac{(x-x^*)^2}{2\sigma^2}\right)$$

true value x*

Maximizing this likelihood is equivalent to minimizing the sum of squared errors:

$$\sum_i \|x_i - x_i^*\|_2^2$$

CONVEX-CONCAVE RATIO

$$\text{CONVEX-CONCAVE RATIO} = \frac{\text{CONVEX FUNCTION F}}{\text{CONCAVE FUNCTION G}}$$

$$f = \frac{(aX)^2+(bX)^2}{p_3 X}$$

$$g = p_3 X$$

$$\|r\|_2^2 = \frac{(aX)^2 + (bX)^2}{(p_3 X)^2}$$

r - reprojection error.

a,b- vectors that depend on camera matrix

$$\min_x \sum_{i=1}^{p} \frac{f_i(x)}{g_i(x)} \quad \text{subject to} \quad x \in D$$

SEDUMI

- Matlab/GNU Octave package for solving convex optimization problems involving linear equations and inequalities, second-order cone constraints, and semidefinite constraints
- Handles very large-scale problems and remains robust even when the problem data is ill-conditioned.
- SeDuMi is often used with CVX, a modeling system for convex optimization in Matlab and is Open-Source

CVXPY

- Python-embedded modeling language for convex optimization problems.
- It automatically transforms the problem into standard form, calls a solver, and unpacks the results.
- Support for Convex Problem Classes:
  Linear Programming, Quadratic Programming, Second-Order Cone Programming

cvxpy status

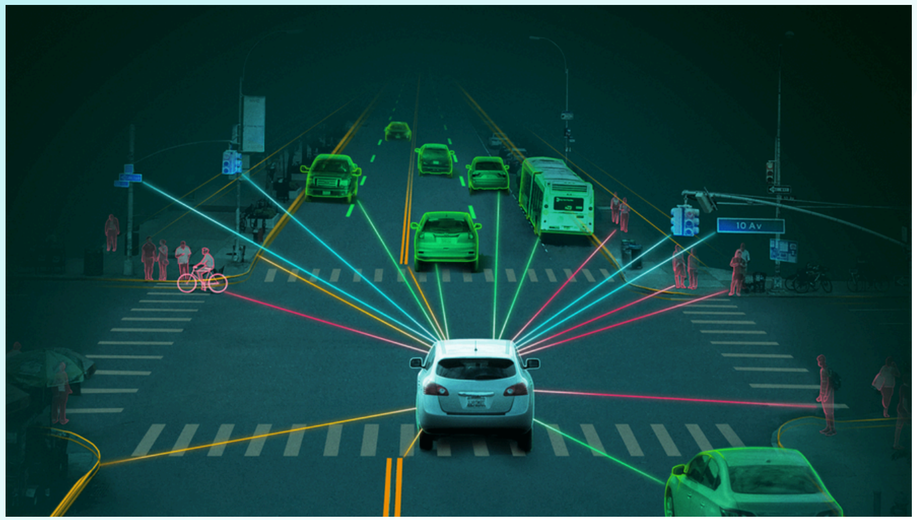
- optimal: An optimal solution that meets all constraints.
- infeasible: No solution that satisfies all constraints
- unbounded: No finite optimal value exists

EXTENSION

Real-Time Requirements for Autonomous Systems

Challenge: Autonomous systems like drones and self-driving cars require real-time triangulation, which demands high computational efficiency.

Use Case: Real-time 3D mapping in autonomous vehicles, where efficient triangulation is crucial for on-the-fly decisions



- In dynamic triangulation, where points of interest may be moving (e.g., tracking vehicles, pedestrians, or other objects in motion), branch-and-bound helps manage the complexity by refining bounds dynamically and focusing on likely regions for the correct solution.

- Use a Kalman filter for each tracked point to predict its next location based on previous frames. This helps estimate the likely position of each point in the current frame.

BENEFITS OF BRANCH & BOUND THEORY IN DYNAMIC TRIANGULATION

- **Reduced Search Space:**

By only focusing on likely regions based on motion prediction, B&B minimizes unnecessary calculations.

- **Higher Efficiency:**

Pruning unpromising branches reduces computational load, which is critical for real-time applications.

- **Improved Accuracy:**

Iterative refinement within probable regions ensures that the 3D point estimates remain accurate, even as objects move.

OUR TEAM

Sirimanna
N.T.W.

210610H

AMARASINGHE
A.A.W.L.R.

210031H

THANK YOU!