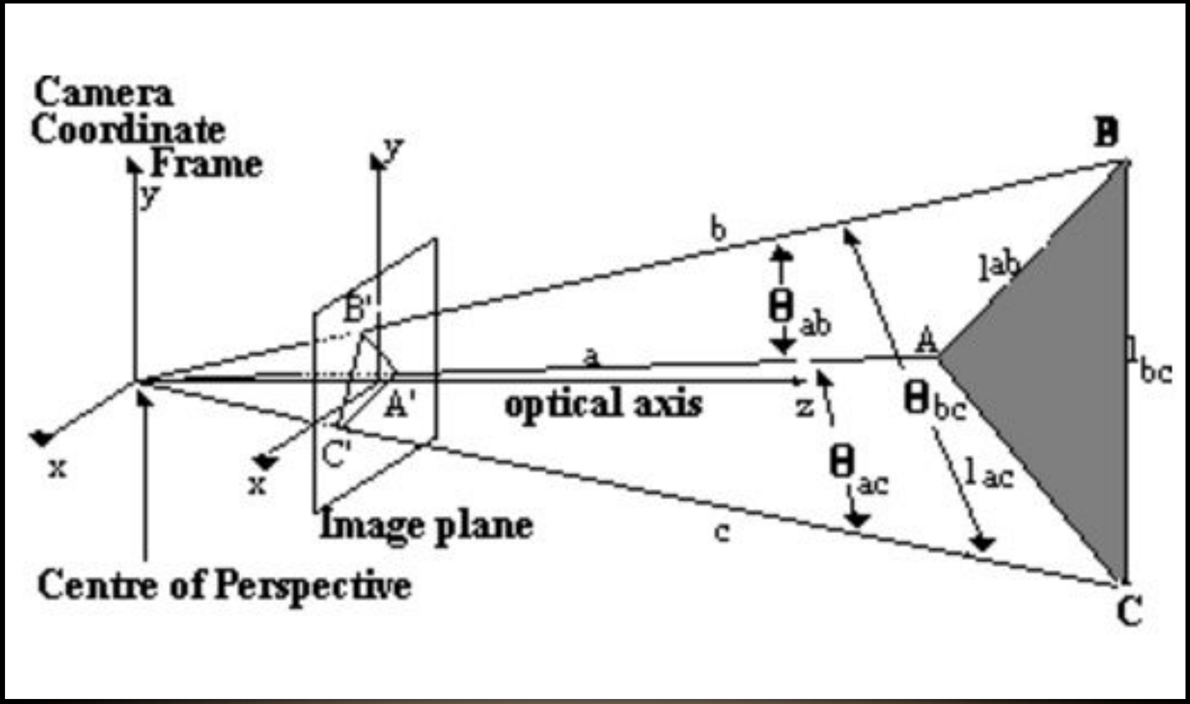
PNP

Perspective-n-Point

Почежерцев А.Г.
Гр. 43328/1
2016 г.

ОПРЕДЕЛЕНИЕ

ЗАДАЧА PnP – восстановление точек в 3D-пространстве по их перспективной проекции на плоскость сенсора камеры.



СУТЬ

Априорно известны:

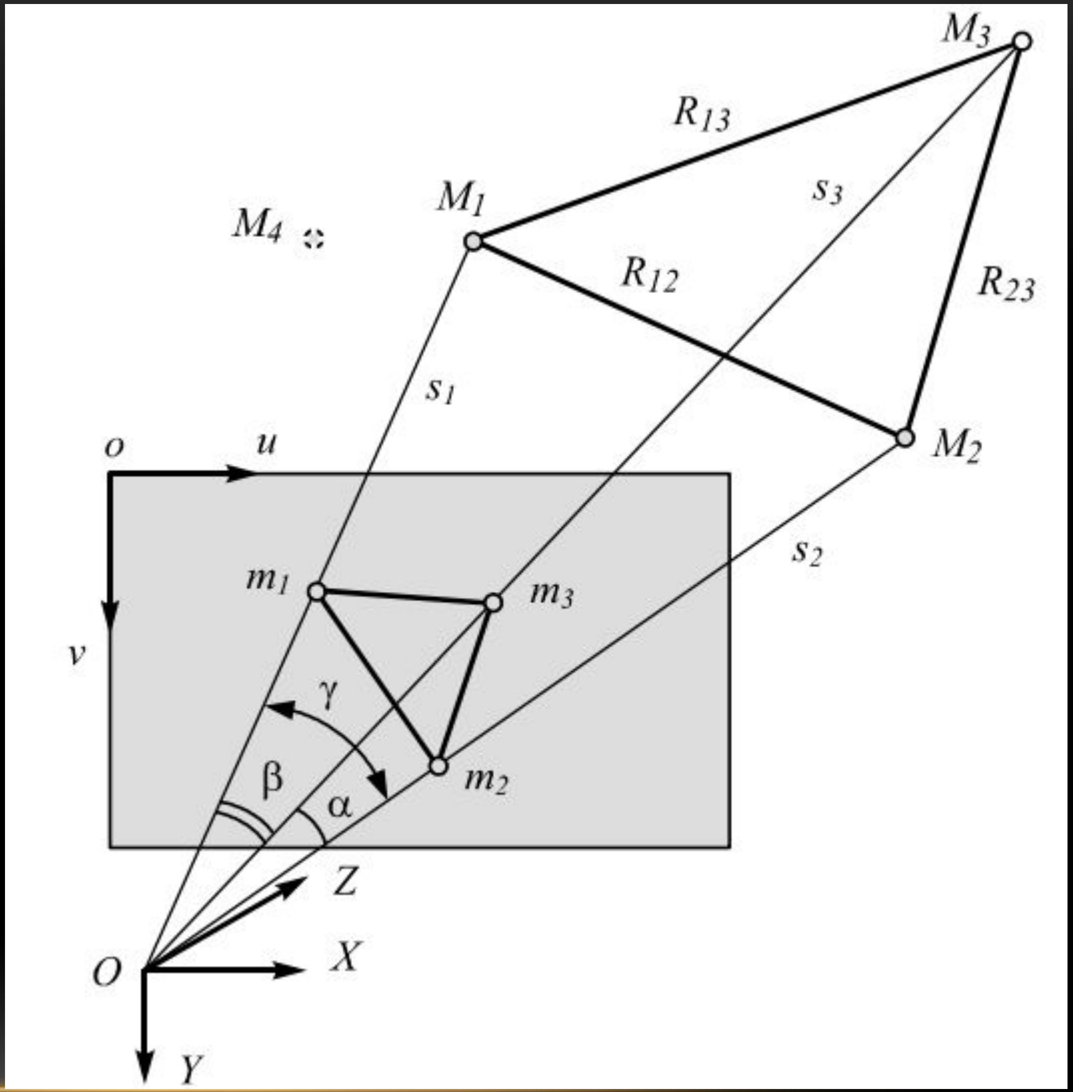
- R_{ij} (то есть геометрия тела)
- Коэффициенты дисторсии
- Матрица внутренних параметров камеры
- Координаты m1, m2, m3

Ищем в процессе:

- α, β, γ

Ищем в итоге:

- S1, S2, S3



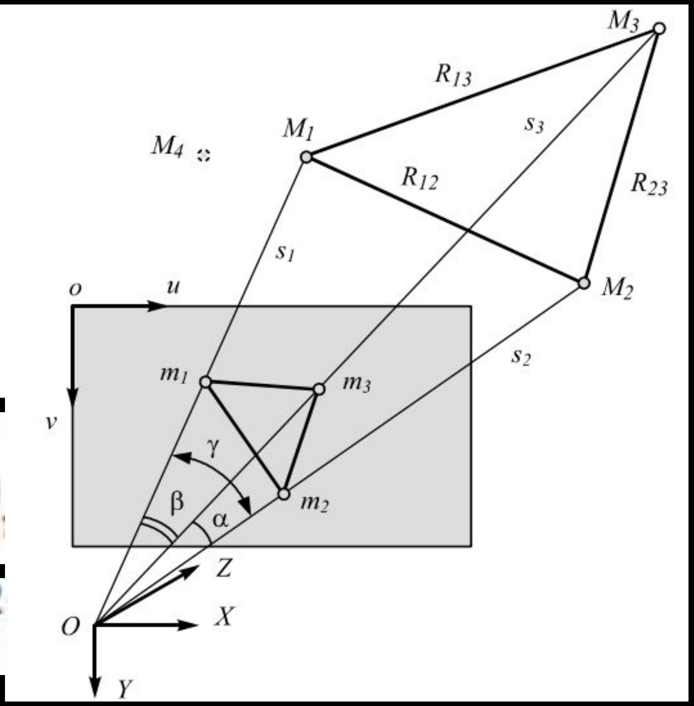
МАТЕМАТИКА P3P

Система уравнений:

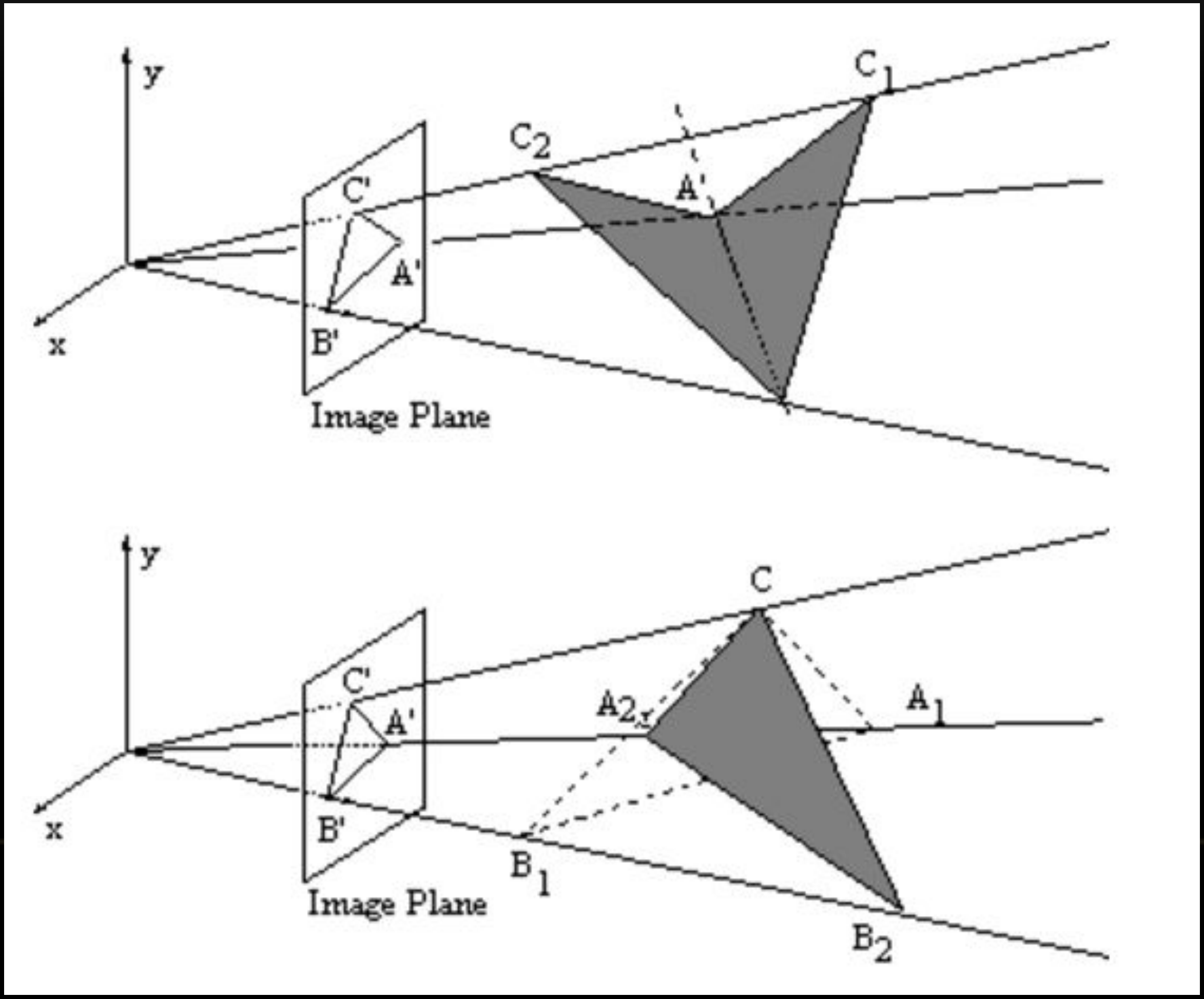
$$\begin{cases} s_2^{\,2} + s_3^{\,2} - 2s_2 s_3 \cos\alpha - R_{23}^{\,2} = 0, \\ s_1^{\,2} + s_3^{\,2} - 2s_1 s_3 \cos\beta - R_{13}^{\,2} = 0, \\ s_1^{\,2} + s_2^{\,2} - 2s_1 s_2 \cos\gamma - R_{12}^{\,2} = 0, \end{cases}$$

$$a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0 = 0$$

$$t = s_1^{\,2}$$



ПРОБЛЕМЫ P3P



ОДНОЗНАЧНОЕ РЕШЕНИЕ ДАЕТСЯ ПРИ N≥4

$$\begin{cases} a_4 t^4 + a_3 t^3 + a_2 t^2 + a_1 t + a_0 = 0, \\ a_4^* t^4 + a_3^* t^3 + a_2^* t^2 + a_1^* t + a_0^* = 0, \\ a_4^{**} t^4 + a_3^{**} t^3 + a_2^{**} t^2 + a_1^{**} t + a_0^{**} = 0, \end{cases}$$

$$\mathbf{A}_x \mathbf{t}_x = \mathbf{0},$$

где $\mathbf{t}_x = [1, t, t^2, t^3, t^4]^{\mathrm{T}}$, $t_i = t^i$, $i = \overline{0,4}$, $\mathbf{0} = [0, 0, 0]^{\mathrm{T}}$,

$$\mathbf{A}_x = \begin{bmatrix} a_0 & a_1 & a_2 & a_3 & a_4 \\ a_0^* & a_1^* & a_2^* & a_3^* & a_4^* \\ a_0^{**} & a_1^{**} & a_2^{**} & a_3^{**} & a_4^{**} \end{bmatrix}.$$

РЕШЕНИЕ В OPENCV

```
bool cv::solvePnP ( InputArray    objectPoints,
                    InputArray    imagePoints,
                    InputArray    cameraMatrix,
                    InputArray    distCoeffs,
                    OutputArray   rvec,
                    OutputArray   tvec,
                                  useExtrinsicGuess =
                    bool          false,
                    int           flags = SOLVEPNP_ITERATIVE
                  )
```

АЛГОРИТМЫ

- SOLVEPNP_ITERATIVE// На базе метода Левенберга-Марквардта
- SOLVEPNP_P3P // В действительности используется 4 точки*
- SOLVEPNP_EPNP // Efficient Perspective-n-Point
- SOLVEPNP_DLS // A Direct Least-Squares
- SOLVEPNP_UPNP // Uncalibrated PnP, "Exhaustive Linearization for Robust Camera Pose and Focal Length Estimation"

* - http://docs.opencv.org/3.1.0/d9/d0c/group__calib3d.html#ga549c2075fac14829ff4a58bc931c033d

EPNP

$$p_i^w = \sum_{j=1}^{4} \alpha_{ij} c_j^w$$

$$p_i^c = \sum_{j=1}^{4} \alpha_{ij} c_j^c$$

$$\sum_{j=1}^{4} \alpha_{ij} = 1$$

$$s_i\, p_i^c = K \sum_{j=1}^{4} \alpha_{ij} c_i^c.$$

$$c_j^c = \begin{bmatrix} x_j^c & y_j^c & z_j^c \end{bmatrix}^T$$

$$\sum_{j=1}^{4} \alpha_{ij} f_x x_j^c + \alpha_{ij}(u_0 - u_i) z_j^c = 0$$

$$\sum_{j=1}^{4} \alpha_{ij} f_y y_j^c + \alpha_{ij}(v_0 - v_i) z_j^c = 0.$$

$$x = \begin{bmatrix} c_1^{c^T} & c_2^{c^T} & c_3^{c^T} & c_4^{c^T} \end{bmatrix}^T$$

$$x = \sum_{i=1}^{N} \beta_i v_i$$

RANSAC (БОЛЕЕ УСТОЙЧИВОЕ К ВЫБРОСАМ)

```
bool cv::solvePnPRansac ( InputArray    objectPoints,
                          InputArray    imagePoints,
                          InputArray    cameraMatrix,
                          InputArray    distCoeffs,
                          OutputArray   rvec,
                          OutputArray   tvec,
                          bool          useExtrinsicGuess = false,
                          int           iterationsCount = 100,
                          float         reprojectionError = 8.0,
                          double        confidence = 0.99,
                          OutputArray   inliers = noArray(),
                          int           flags = SOLVEPNP_ITERATIVE
                        )
```

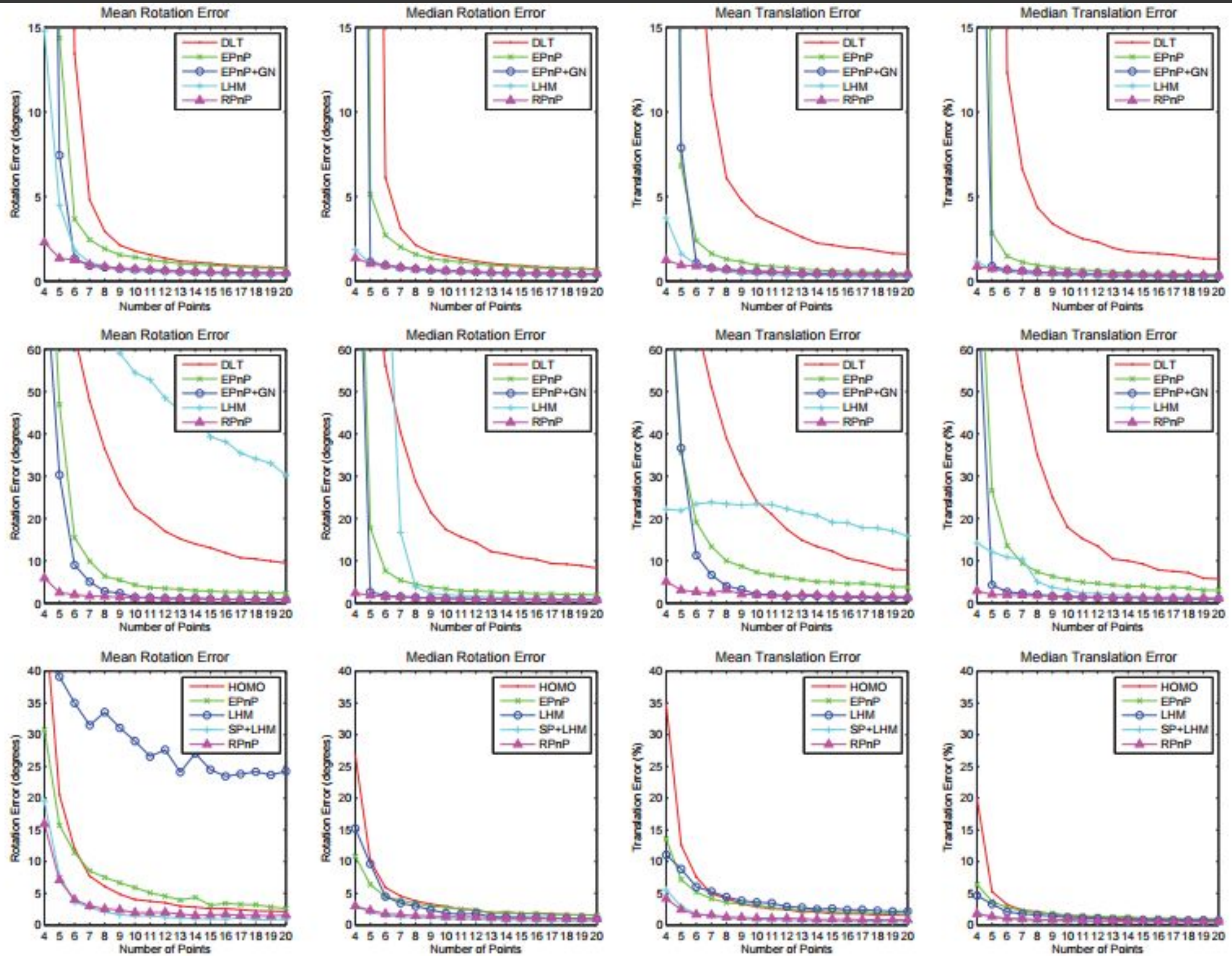
О МЕТОДАХ:

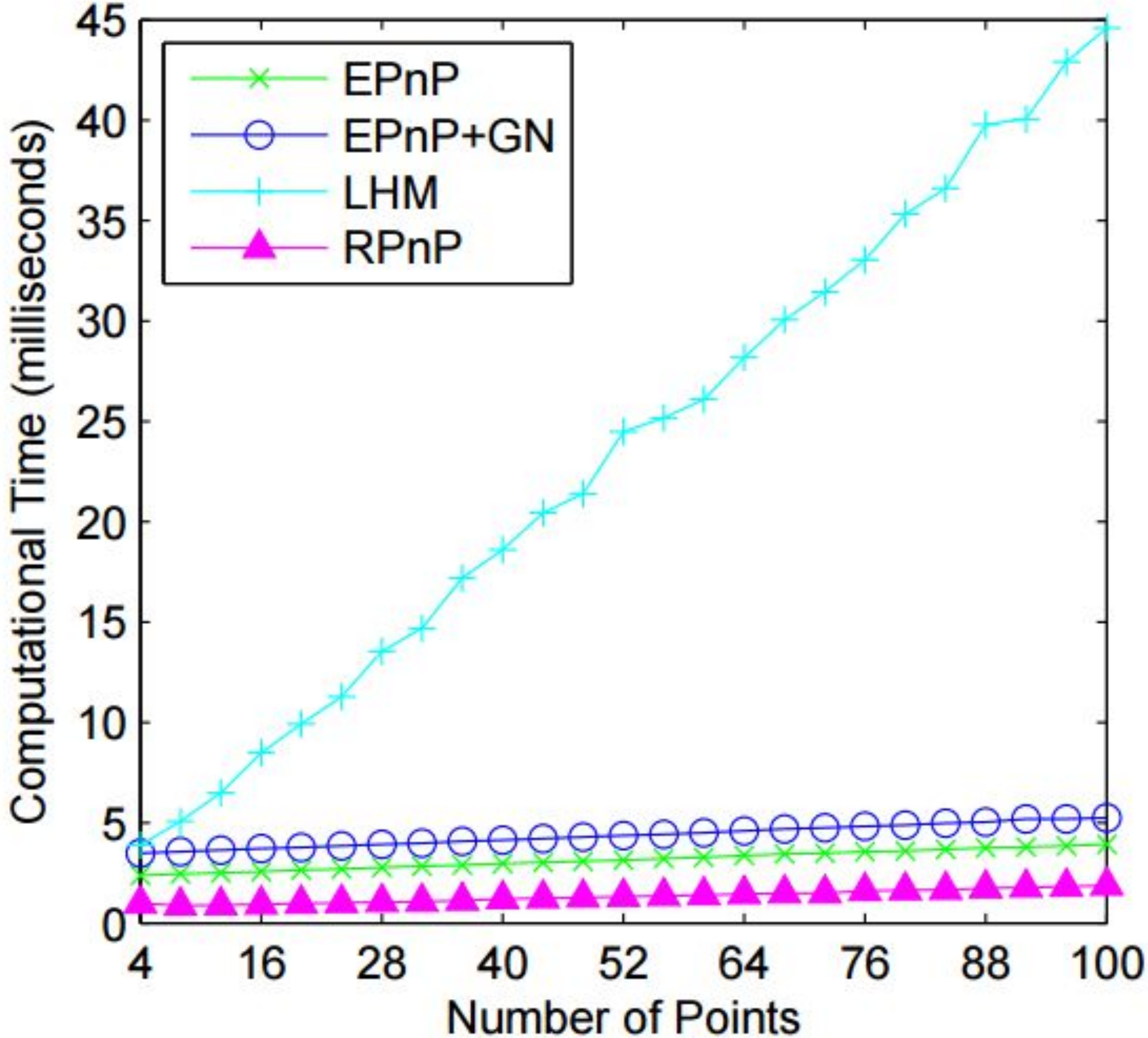
- DLT - direct linear transformation method
- EPnP - efficient O(n) non-iterative solution of PnP
- EPnP+GN - the EPnP method followed with a Gaussian-Newton optimizer
- LHM - Lu et al method
- HOMO - the homography method for planar targets

Ordinary 3D Case
$n = 4, 5, \cdots 20$
$\sigma = 3$ *pixels*

Quasi-Singular Case
$n = 4, 5, \cdots 20$
$\sigma = 3$ *pixels*

Planar Case
$n = 4, 5, \cdots 20$
$\sigma = 3$ *pixels*





EPnP
EPnP+GN
LHM
RPnP
Computational Time (milliseconds)
45
40
35
30
25
20
15
10
5
0
4
16
28
40
52
64
76
88
100
Number of Points

ПРИМЕР РЕШЕНИЯ ЗАДАЧИ PNP

ИСТОЧНИКИ

- http://docs.opencv.org/3.1.0/d9/d0c/group__calib3d.html#ga549c2075fac14829ff4a58bc931c033d
- http://xuchi.weebly.com/uploads/5/6/7/3/5673896/rpnp.pdf
- http://www.computeroptics.smr.ru/KO/PDF/KO39-3/390317.pdf
- https://en.wikipedia.org/wiki/Perspective-n-Point
- http://upcommons.upc.edu/bitstream/handle/2117/22931/1404-Exhaustive-linearization-for-robust-camera-pose-and-focal-length-estimation.pdf?sequence=1