# Optimization method to solving relative rotation and translation of galvonometer systems

Owen Lu
Electroimpact Inc.
owenl@electroimpact.com

***Abstract*—This document describes one method to solving for the rotation and translation of the galvanometer axes relative to a reference coordinate frame. Optimization methods are suggested due to the current algorithm returning different results based on the registration order of calibration points.**

## I. Introduction

At the core of finding the rotation and translation of the laser reference frame in respect to the absolute reference frame, ray equations must be solved. Loosely speaking, the simplified problem has a set of 4 or more angle pairs $\theta_1, \theta_2$ describing mirror angles with corresponding calibration points $x$ in 3 space for which a laser passes through. The ray, described by a mathematical line corresponding $\theta_1, \theta_2$ should then pass through all calibration points if no error exists.

In experience, the laser projectors solving method gives results that depend greatly on the order in which points are registered. This is not ideal since consistent results are unable to be realized. Furthermore, there should be a "best" solution that gives the least error possible, and the operator should not need to permute calibration points to find this best solution.

## II. Quarternion rotation

Quaternion representation of rotation is well understood and resources can be found in many places online. However, a brief introduction here will be done for convenience and notation purposes.

Suppose that we wish to generate rotation of a vector around $\bar{u}$ by angle $\theta$. This can be done using a quaternion $q$ and generating a rotation matrix $\boldsymbol{R}$.

$$q = \left[\cos\left(\frac{\theta}{2}\right), u_x \sin\left(\frac{\theta}{2}\right), u_y \sin\left(\frac{\theta}{2}\right), y_z \sin\left(\frac{\theta}{2}\right)\right]'$$

$$q = [q_r, q_i, q_j, q_k]'$$

Where

$$\boldsymbol{R}(q) = \begin{bmatrix} 1-2q_j^2-2q_k^2 & 2(q_iq_j - q_kq_r) & 2(q_iq_k + q_jq_r) \\ 2(q_iq_j + q_kq_r) & 1-2q_i^2-2q_k^2 & 2(q_jq_k - q_iq_r) \\ 2(q_iq_k - q_jq_r) & 2(q_jq_k + q_iq_r) & 1-2q_i^2-2q_j^2 \end{bmatrix}$$

Then applying the rotation to general vector $\bar{v}$ to formulate rotated vector $\bar{v}'$ can be done with the equation following.

$$\bar{v}' = \boldsymbol{R}(q)\bar{v}$$

## III. Ray formulation

In a previous document ray equations in the galvanometer frame were derived depending on mirror angles. The results are reused below without derivation for convenience.

$$\bar{p}' = \begin{pmatrix} x_1 - y_1\cot(\theta_1) - \dfrac{t_1}{2\sin(\theta_1)} \\ 0 \\ 0 \end{pmatrix}$$

$$\bar{p}'' = \bar{p}' + \begin{pmatrix} \left(y_2 - z_2\cot(\theta_2) - \dfrac{t_2}{2\sin(\theta_2)}\right)\cot(2\theta_1) \\ y_2 - z_2\cot(\theta_2) - \dfrac{t_2}{2\sin(\theta_2)} \\ 0 \end{pmatrix}$$

$$\bar{u}'' = \begin{pmatrix} \cos(2\theta_1) \\ \sin(2\theta_1)\cos(2\theta_2) \\ \sin(2\theta_1)\sin(2\theta_2) \end{pmatrix}$$

The ray in galvanometer coordinates, defined as frame B is given below.

$$r_{i_B} = \bar{p}''(\theta_1, \theta_2) + \lambda\bar{u}''(\theta_1, \theta_2)$$

## IV. Ray transformation

The ray equations must be converted to equations in the reference frame since calibration points are known in said frame.

Let $L$ be the translation of $B$ to $A$.

$$\bar{l} = [l_1, l_2, l_3]'$$

Let $q$ be the quaternion describing the rotation of $B$ to $A$

$$q = [q_r, q_i, q_j, q_k]'$$

Suppose a line $r$ is defined in the equation below in the galvanometer frame $B$.

$$r_B\colon \bar{p}'' + \lambda\bar{u}'' \tag{1}$$

If we wish to find the same line equation within reference frame $A$ we must know the quaternion $q$ and translation $\bar{l}$ that when applied to $A$ creates an equivalent frame with $B$.

At this point the line becomes a straight forward calculation.

$$r_A\colon \boldsymbol{R}(q)\bar{p}'' + \bar{l} + \lambda\boldsymbol{R}(q)\bar{u}''$$

Let

$$\boldsymbol{R}(q)\bar{u}'' = \bar{u}_A$$

$$\bar{x}_{0_A} = \boldsymbol{R}(q)\bar{p}'' + \bar{l}$$

By substitution

$$r_A = \bar{x}_{0_A} + \lambda\bar{v}_A$$

## V. Objective formulation

Ideally the objective has a physical meaning so that the user can easily verify whether the optimization algorithm has converged to a reasonable solution. For this reason, the average distance from the ray to a calibration point is used.

The perpendicular distance between some point $\bar{x}_i$ and line $r_i$ can be given by the equation below.

$$dist(r_i, x_i) = norm\left( (\bar{x}_i - \bar{x}_{0_A}) - \left( (\bar{x}_i - \bar{x}_{0_A}) \cdot \bar{v}_A \right) \bar{v}_A \right)$$

The average distance to a calibration $M$ is given below.

$$M = \frac{1}{n}\sum_{i=1}^{n} dist(r_{i_A}(q, \bar{l}), \bar{x}_i)$$

Under constraint

$$\|q\| = 1$$

## VI. Numerical testing

To test whether this method can converge upon reasonable results, a sample data set was used assuming the following parameters.

$$k = 1$$

$$t_1 = 5mm$$

$$t_2 = 5mm$$

$$x_1 = 0$$

$$y_1 = 0$$

$$y_2 = 40mm$$

$$z_2 = 0$$

| $\theta_1\ (deg)$ | $\theta_2\ (deg)$ | $x(mm)$ | $y(mm)$ | $z(mm)$ |
|---|---|---|---|---|
| 45 | 55 | $-3.53$ | $-327.02$ | 1000 |
| 45 | 65 | $-3.53$ | $-801.85$ | 1000 |
| 55 | 55 | $-403.82$ | $-327.02$ | 1000 |
| 55 | 65 | $-491.73$ | $-801.85$ | 1000 |
| 65 | 55 | $-926.71$ | $-327.02$ | 1000 |
| 65 | 65 | $-1129.37$ | $-801.85$ | 1000 |

The optimization function used was $fmincon$ in MATLAB. Random seed values were used with to generate quaternions rotations within 10 degrees of the correct solution, while errors of 0.5 metres was used for the translational error.

A plot of the average error as a function of the seed angular error and the seed displacement error is given following.

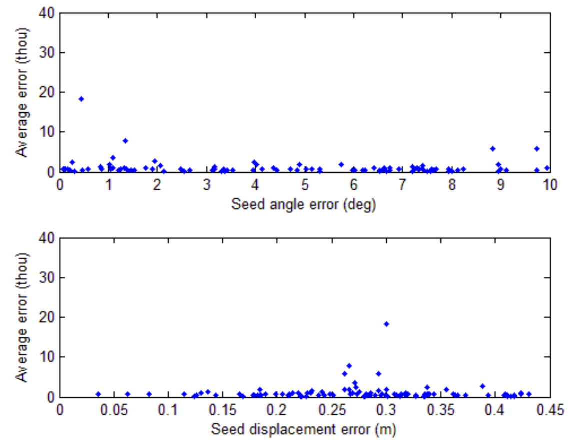


Figure 1 – Average distance error vs. Seed angle error (top); Average distance error vs. Seed displacement error (bottom

From the results, the error is small, under 20 thou if the initial seed can be estimated to reasonable accuracy. More tests are necessary, but initial feasibility has been validated.

*Note: A key setting in the MATLAB function is that TolFun = 0.000001.