My version of the notation in [1]. I can not believe that I am making this for a workshop paper!

All vectors are assumed to be column vectors, hence

$$x = (x_1, x_2, x_3)$$

is the same as

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}.$$

I **don't** use bold characters to denote vectors and matrices, hence x_1, x_2, x_3 could be vectors as well. I use Σ instead of Ω (which is used in the paper), because I want $\sigma \in \Sigma$, and I use ω for angular velocity.

1 Notation

We are interested in a subset Σ of the image plane (*i.e.*, $\Sigma \subset \mathbb{R}^2$).

- Let I_i be a function that assigns a gray-scale level to each element of Σ for the i^{th} image, or formally

$$I_i : \Sigma \to [0, 1].$$

- Let h_i be a function that assigns a depth (in meters) to each element of Σ for the i^{th} image, or formally

$$h_i : \Sigma \to \mathbb{R}_+.$$

- Let S_i be a (vector-valued) function that defines a surface given the i^{th} image, or formally

$$S_i : \Sigma \to \mathbb{R}^3,$$

$$S_i(\sigma) = \begin{bmatrix} \frac{\sigma_1 + c_1}{c_3} h_i(\sigma) \\ \frac{\sigma_2 + c_2}{c_4} h_i(\sigma) \\ h_i(\sigma) \end{bmatrix},$$

where $\sigma = (\sigma_1, \sigma_2) \in \Sigma$, and $c_1, c_2, c_3, c_4 \in \mathbb{R}$ are camera parameters. In addition (for convenience) we define

$$\bar{S}_i(\sigma) = \begin{bmatrix} S_i(\sigma) \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{\sigma_1 + c_1}{c_3} h_i(\sigma) \\ \frac{\sigma_2 + c_2}{c_4} h_i(\sigma) \\ h_i(\sigma) \\ 1 \end{bmatrix}.$$

- Let π_i be a (vector-valued) function that projects 3D points to the image plane, or formally

$$\pi : \mathbb{R}^3 \to \Sigma,$$

$$\pi(x) = \begin{bmatrix} \frac{x_1}{x_3} c_3 - c_1 \\ \frac{x_2}{x_3} c_4 - c_2 \end{bmatrix},$$

where $x = (x_1, x_2, x_3) \in \mathbb{R}^3$. Note that $\sigma = \pi(S_i(\sigma))$. In addition (for convenience) we define $\bar{\pi} : \mathbb{R}^4 \to \Sigma$,

$$\bar{\pi}(\bar{x}) = \begin{bmatrix} \frac{\bar{x}_1}{\bar{x}_3} c_3 - c_1 \\ \frac{\bar{x}_2}{\bar{x}_3} c_4 - c_2 \end{bmatrix},$$

where $\bar{x} = (x, 1)$. Hence $\sigma = \bar{\pi}(\bar{S}_i(\sigma))$.

- In the subsequent notation it will be assumed that $\sigma \in \mathbb{R}^2$, $x \in \mathbb{R}^3$ and $\bar{x} = (x, 1)$.

- $T \in \mathbb{R}^{4\times 4}$ will denote a homogeneous matrix *i.e.*,

$$T = \begin{bmatrix} R & r \\ 0^T & 1 \end{bmatrix},$$

  where R is 3×3 rotation matrix, $r \in \mathbb{R}^3$ and 0^T is a row vector of zeros (with appropriate dimensions). This is equivalent to saying that $T \in SE(3)$ (but I try to avoid the latter notation - ask me and I will explain why).

- $\xi \in \mathbb{R}^6$ will be used to denote a vector "stacking" linear velocities $v = (v_1, v_2, v_3)$ and angular velocities $\omega = (\omega_1, \omega_2, \omega_3)$

$$\xi = \begin{bmatrix} v \\ \omega \end{bmatrix}.$$

  In addition we define

$$\hat{\xi} = \begin{bmatrix} \tilde{\omega} & v \\ 0^T & 0 \end{bmatrix},$$

  where $\tilde{\omega}$ is a skew-symmetric matrix *i.e.*,

$$\tilde{\omega} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}.$$

  $\hat{\xi}$ is referred to as a *twist*, while ξ is a 6 dimensional vector that parametrizes a twist (see for example [2], pp. 41). Sometimes ξ is called a twist as well.

- A property that we need later on (see [2], pp. 41)

$$e^{\hat{\xi}\Delta t} \quad \text{is a homogeneous matrix,}$$

  where $\Delta t \in \mathbb{R}$ denotes the length of a time interval. Read [2], pp. 42 after equation (2.36) (up to **Proposition 2.9**). Probably you remember that in the Manipulation and Control course you had to prove that $e^{\tilde{\omega}\Delta t}$ is a rotation matrix (the above property is a generalization). See example `example_exp_map.m`.

2 The main idea

I will try to present the main idea in a slightly different way (than in the paper) - but of course I arrive at the same equations in the end. I will put emphasis on the concepts that (I believe) you don't understand. It is worth mentioning that the most important sentence in [1] is "Our key idea is ..." (just before Section 2.1).

3 Problem 1

First, we define a preliminary problem (for the moment let us forget about *photo-consistency*). Given two functions S_1 and S_2 (if you prefer, you can think of them as the surfaces corresponding to two images), we define the distance $\mathbf{dist}(S_1, S_2)$ as

$$\mathbf{dist}(S_1, S_2) = \int_\Sigma \|S_2(\sigma) - S_1(\sigma)\|^2 d\sigma.$$

Clearly $\mathbf{dist}(S_1, S_2)$ defines a *norm*. Note that Σ is a finite set of 2D points, and probably it would be more appropriate to use $\sum_\Sigma$ instead of $\int_\Sigma$.

Imagine that we want to have some way of influencing the distance between the functions S_1 and S_2. At the moment we have 0 Degrees-of-Freedom (DoF), since we can only evaluate $\mathbf{dist}(S_1, S_2)$. The first observation in the paper is that since we interpret S_1 and S_2 as two surfaces in 3D, the assumption that only the camera moves (but the scene stays unchanged) means that S_1 and S_2 are related through a rigid-body transformation. Hence, a reasonable choice for "control actions" would be to introduce the capability to rotate and translate one of the surfaces.

Let us define a new distance function (this time parametrized by a homogeneous matrix T)

$$\mathbf{dist}(S_1, S_2, T) = \int_\Sigma \|T\bar{S}_2(\sigma) - \bar{S}_1(\sigma)\|^2 d\sigma.$$

Now we have 6 DoF. Note that $T\bar{S}_2(\sigma) \in \mathbb{R}^4$ is a vector whose fourth coordinate is equal to 1. Consider the problem

$$\begin{aligned} \underset{T}{\text{minimize}} \quad & \mathbf{dist}(S_1, S_2, T) \\ \text{subject to} \quad & T \text{ is a homogeneous matrix.} \end{aligned} \tag{1}$$

As noted in the paper, this is a non-convex problem. Further note that every parametrization of T (for example using Euler angles or a unit Quaternion) leads to certain complications (I will explain in person).

The second observation in the paper is that since **we know the duration of camera motion Δt** (this is very important) (1) is equivalent to

$$\underset{\xi}{\text{minimize}} \quad \mathbf{dist}(S_1, S_2, \xi) = \int_\Sigma \|e^{\hat{\xi}\Delta t}\bar{S}_2(\sigma) - \bar{S}_1(\sigma)\|^2 d\sigma. \tag{2}$$

Note that there are no constraints because of our choice of parametrization of T.

The third observation in the paper is that, for small camera motion, instead of solving (2) one can solve an approximate problem that can lead to "reasonable" results. Let us denote the first term of the integrand of (2) by

$$f(\sigma, \xi) = e^{\hat{\xi}\Delta t}\bar{S}_2(\sigma).$$

The function $f(\sigma, \xi)$ can be approximated by using the first two terms of its Taylor series expansion (around $\xi = 0$)

$$f(\sigma, \xi) \approx f(\sigma, \xi)\big|_{\xi=0} + \nabla_\xi f(\sigma, \xi)\big|_{\xi=0}\, \xi,$$

where $\nabla_\xi f(\sigma, \xi)$ denotes the gradient of $f(\sigma, \xi)$ w.r.t. ξ, and $\nabla_\xi f(\sigma, \xi)\big|_{\xi=0}$ implies that $\nabla_\xi f(\sigma, \xi)$ is evaluated at $\xi = 0$. Using the fact that e^0 is an identity matrix and setting $J(\sigma) = \nabla_\xi f(\sigma, \xi)\big|_{\xi=0}$ the above equation becomes

$$f(\sigma, \xi) \approx \bar{S}_2(\sigma) + J(\sigma)\xi. \tag{3}$$

Substituting (3) in (2) leads to

$$\underset{\xi}{\text{minimize}} \quad \int_{\Sigma} \|J(\sigma)\xi + d(\sigma)\|^2 d\sigma, \tag{4}$$

where $d(\sigma) = \bar{S}_2(\sigma) - \bar{S}_1(\sigma)$. The integrand of (4) is an affine function of ξ.

3.1 The Jacobian $J(\sigma)$

$$\frac{\partial\big[e^{\hat{\xi}\Delta t}\bar{S}_2(\sigma)\big]}{\partial \xi_i} = \frac{\partial\big[e^{\hat{\xi}\Delta t}\big]}{\partial \xi_i}\bar{S}_2(\sigma) + e^{\hat{\xi}\Delta t}\underbrace{\frac{\partial\big[\bar{S}_2(\sigma)\big]}{\partial \xi_i}}_{0} = \underbrace{\frac{\partial\big[e^{\hat{\xi}\Delta t}\big]}{\partial \xi_i}}_{\in\mathbb{R}^{4\times 4}}\bar{S}_2(\sigma) \in \mathbb{R}^4, \quad i = 1, \ldots, 6.$$

Hence, $J(\sigma) \in \mathbb{R}^{4\times 6}$, which makes sense since $\xi \in \mathbb{R}^6$ and $d(\sigma) \in \mathbb{R}^4$. We will generate a C function that forms $J(\sigma)$ using Maple (or Matlab) (it is very simple).

3.2 The normal equations

Equation (4) can be expressed as

$$\underset{\xi}{\text{minimize}} \quad \int_{\Sigma} \big[J(\sigma)\xi + d(\sigma)\big]^T \big[J(\sigma)\xi + d(\sigma)\big] d\sigma,$$

which after some algebraic manipulations can be put in the following form

$$\underset{\xi}{\text{minimize}} \quad \int_{\Sigma} \big[\xi^T J(\sigma)^T J(\sigma)\xi + 2\xi^T J(\sigma)^T d(\sigma) + d(\sigma)^T d(\sigma)\big] d\sigma.$$

Let $\Sigma = \{\sigma_1, \ldots, \sigma_N\}$, *i.e.*, Σ has N elements. Define

$$H = \sum_{i=1}^{N} J(\sigma_i)^T J(\sigma_i) \in \mathbb{R}^{6\times 6}.$$

Clearly $H(\sigma)$ is a symmetric matrix. Furthermore, it is very likely for it to be positive definite (why).

$$g = \sum_{i=1}^{N} J(\sigma)^T d(\sigma) \in \mathbb{R}^6.$$

Hence, by noting that the term $d(\sigma)^T d(\sigma)$ does not depend on ξ, we obtain the following problem

$$\underset{\xi}{\text{minimize}} \quad \xi^T H \xi + 2\xi^T g. \tag{5}$$

Hence, (assuming $H > 0$) $\xi^\star = -H^{-1}g$.

3.3 Concluding remarks

You could see the whole procedure in the case when we disregard the information about gray-level. The most important point that you should note in the next section is the way we reformulate the problem so that we are able to account for the additional information we have. I mention this because I know that this is of interest to you. Furthermore, note that we already used some additional information (the time elapsed between two images).

4 Problem 2

We want to use the additional information encoded in $I_i(\sigma)$. Let us define the distance $\mathbf{dist}(I_1, I_2)$ between the two functions I_1 and I_2 as

$$\mathbf{dist}(I_1, I_2) = \int_\Sigma \|I_2(\sigma) - I_1(\sigma)\|^2 d\sigma.$$

The key point in the paper is that we can write the above distance as

$$\mathbf{dist}(I_1, I_2) = \int_\Sigma \|I_2(\bar{\pi}(\bar{S}_2(\sigma))) - I_1(\sigma)\|^2 d\sigma,$$

again noting that $\sigma = \bar{\pi}(\bar{S}_2(\sigma))$. Hence, (as in the previous section)

$$\mathbf{dist}(I_1, I_2, T) = \int_\Sigma \|I_2(\bar{\pi}(T\bar{S}_2(\sigma))) - I_1(\sigma)\|^2 d\sigma.$$

Or equivalently

$$\mathbf{dist}(I_1, I_2, \xi) = \int_\Sigma \|I_2(\bar{\pi}(e^{\hat{\xi}\Delta t}\bar{S}_2(\sigma))) - I_1(\sigma)\|^2 d\sigma.$$

4.1 Taking it easy

Next, I make an intermediate step, so that you clearly see the procedure. Let us try to minimize

$$\int_\Sigma \left[\bar{\pi}(\underbrace{e^{\hat{\xi}\Delta t}\bar{S}_2(\sigma)}_{f(\sigma,\xi)}) - \sigma\right]^2 d\sigma.$$

Essentially, again we disregard the additional information (*i.e.*, gray-scale) and do something similar to the procedure in Section 3 (but this time we have the additional function $\bar{\pi}$).

Let us now linearize $\bar{\pi}(f(\sigma,\xi))$. Using the chain rule we obtain

$$\frac{\partial\left[\bar{\pi}(f(\sigma,\xi))\right]}{\partial\xi} = \frac{\partial\bar{\pi}}{\partial f}\frac{\partial f}{\partial\xi}.$$

Now, we already know $\frac{\partial f}{\partial \xi} \in \mathbb{R}^{4\times 6}$. Computing $\frac{\partial\bar{\pi}}{\partial f} \in \mathbb{R}^{2\times 4}$ is straightforward (but I am too lazy)

```
syms f1 f2 f3 f4 c1 c2 c3 c4

pi_bar = [f1/f3*c3 - c1; f2/f3*c4 - c2];
f = [f1;f2;f3;f4];

J = jacobian(pi_bar,f);
```

$$\frac{\partial\bar{\pi}}{\partial f} = \begin{bmatrix} \frac{c_3}{f_3} & 0 & -\frac{f_1}{f_3^2}c_3 & 0 \\ 0 & \frac{c_4}{f_3} & -\frac{f_2}{f_3^2}c_4 & 0 \end{bmatrix}.$$

Of course, what I would do in practise is to dump directly

$$\frac{\partial\bar{\pi}}{\partial f}\frac{\partial f}{\partial\xi}$$

in Maple (or Matlab), which will generate a highly optimized C code that forms the 2×6 Jacobian matrix I need. Once the Jacobian is available the procedure is like the one in Section 3.

4.2 The last step

Essentially, the last step would be to find the partial derivative of

$$I_2(\bar{\pi}(f(\sigma,\xi)))$$

w.r.t ξ. Using the chain rule we obtain

$$\frac{\partial\big[I_2(\bar{\pi}(f(\sigma,\xi)))\big]}{\partial\xi} = \frac{\partial I_2}{\partial\bar{\pi}}\frac{\partial\bar{\pi}}{\partial f}\frac{\partial f}{\partial\xi}.$$

Now, here is the bitch. We can easily find an analytical expression for

$$\frac{\partial\bar{\pi}}{\partial f}\frac{\partial f}{\partial\xi},$$

however, since the function I_2 is actually a look-up table, the partial derivative $\frac{\partial I_2}{\partial\bar{\pi}}$ has to be computed numerically (by using finite differences).

4.3 Concluding remarks

Computing derivatives based on finite differences directly in the 6D space of ξ is crazy slow. Of course you could do it, and it should work. When you compute the derivatives, you should use $\xi = 0$ (because you linearize around this point). It would be much better to analytically compute $\frac{\partial\bar{\pi}}{\partial f}\frac{\partial f}{\partial\xi}$, and form $\frac{\partial I_2}{\partial\bar{\pi}}$ numerically.

5 Brainstorming

5.1 idea 1

What if we define

$$S_i(\sigma) = \begin{bmatrix} \frac{\sigma_1+c_1}{c_3}h_i(\sigma)I_i(x) \\ \frac{\sigma_2+c_2}{c_4}h_i(\sigma)I_i(x) \\ h_i(\sigma)I_i(x) \end{bmatrix}, \quad \text{or} \quad S_i(\sigma) = \begin{bmatrix} \frac{\sigma_1+c_1}{c_3}h_i(\sigma)+I_i(x) \\ \frac{\sigma_2+c_2}{c_4}h_i(\sigma)+I_i(x) \\ h_i(\sigma)+I_i(x) \end{bmatrix}.$$

Or something in this spirit. Then we would not even need the function π_i (I will explain later). Essentially I would like to avoid numerical differentiation if possible.

5.2 idea 2

We could linearize a number of times in a row (the second time around $\xi^\star$).

6 Formulation as a dynamical system

Recall that in Section 3 we discussed the following problem

$$\underset{\xi}{\text{minimize}} \;\; \mathbf{dist}(S_1, S_2, \xi) = \int_\Sigma \| e^{\hat{\xi}\Delta t} \bar{S}_2(\sigma) - \bar{S}_1(\sigma) \|^2 d\sigma.$$

Let us use a shorthand notation $\bar{s}_i = \bar{S}_2(\sigma_i)$, *i.e.*, $s_i \in \mathbb{R}^3$ is the 3D point corresponding to σ_i, and $\bar{s}_i = (s_i, 1)$. Then, we observe that

$$\bar{s}_i(\Delta t) = e^{\hat{\xi}\Delta t} \bar{s}_i(0), \tag{6}$$

where $\bar{s}_i(0)$ denotes $\bar{s}_i$ at time $t = 0$, while $\bar{s}_i(\Delta t)$ denotes $\bar{s}_i$ at time $t = \Delta t$. Clearly, (6) is the solution of the following linear (vector) differential equation

$$\dot{\bar{s}}_i = \hat{\xi} \bar{s}_i. \tag{7}$$

Or equivalently,

$$\dot{s}_i = \tilde{\omega} s_i + v. \tag{8}$$

Given N points of interest (*i.e.*, $|\Sigma| = N$), we have

$$\underbrace{\begin{bmatrix} \dot{s}_1 \\ \vdots \\ \dot{s}_N \end{bmatrix}}_{\dot{z}} = \underbrace{\begin{bmatrix} \tilde{\omega} s_1 + v \\ \vdots \\ \tilde{\omega} s_N + v \end{bmatrix}}_{g(z,\xi)},$$

where we used $z = (s_1, \ldots, s_N) \in \mathbb{R}^{3N}$.

To summarize, we have the following dynamical system

$$\dot{z} = g(z, \xi), \quad z \in \mathbb{R}^{3N}, \quad \xi \in \mathbb{R}^6, \quad g : \mathbb{R}^{3N} \times \mathbb{R}^6 \to \mathbb{R}^{3N}. \tag{9}$$

z is the state, while ξ are parameters (that we will find, so that we minimize a given criterion). Define (for future use) the following output equation

$$y = e(z), \quad y \in \mathbb{R}^m, \quad g : \mathbb{R}^6 \to \mathbb{R}^m. \tag{10}$$

Probably at this point you can guess that we will use the output equation in order to account for the additional information we have. In the context of Section 4, $m = 2$.

With the above formulation, we don't even have to "break a sweat", as we can pose the registration problem as a very standard parameter estimation problem. This means that we can just dump it into a dedicated tool that can exploit the structure of the problem (we don't need to implement the iterative optimization scheme by ourselves). I envision to use ACADO. Bye the way, ACADO uses AD gracefully.

References

[1] F. Steinbrücker J. Sturm. D. Cremers, “Real-Time Visual Odometry from Dense RGB-D Images,” *In Workshop on Live Dense Reconstruction with Moving Cameras at the Intl. Conf. on Computer Vision (ICCV),* 2011.

[2] R. Murray, Z. Li, S. Sastry, “A Mathematical Introduction to Robotic Manipulation”, CRC Press, 1994.