Active vision for pose estimation applied to singularity avoidance in visual servoing

Don Agravante, François Chaumette

Active vision for pose estimation applied to singularity avoidance in visual servoing

Don Joven Agravante[1], and François Chaumette[1]

***Abstract*— In active vision, the camera motion is controlled in order to improve a certain visual sensing strategy. In this paper, we formulate an active vision task function to improve pose estimation. This is done by defining an optimality metric on the Fisher Information Matrix. This task is then incorporated into a weighted multi-objective optimization framework. To test this approach, we apply it on the three image point visual servoing problem which has a degenerate configuration - a singularity cylinder. The simulation results show that the singular configurations of pose estimation are avoided during visual servoing. We then discuss the potential of active vision to be integrated into more complex multi-task frameworks.**

***Index Terms*— Visual servoing, optimization and optimal control, probability and statistical methods.**

I. Introduction and background

Visual servoing is a motion control modality that closes the loop with visual information [1]. Because of this, it is very well suited to controlling the gaze direction of robots. For example, we have shown its application to humanoid robots in our previous works [2], [3]. The formulations in [2], [3] allows the gaze to be defined as a reference in image space. Naturally, the reference relates to the objects of interest (e.g., hand and object to be grasped). However, in most cases this type of control only serves to keep the object inside the field-of-view (FoV) or to center them in the image. Although this can already be termed as *active* in the sense that the camera FoV is effectively increased, we would like to improve on this. In particular, we want to control the gaze such that it explicitly improves the performance of the vision algorithms we are using, such as pose estimation. The motivation to do this comes from using the model based tracker of [4], which gives us an estimate of the object pose in our grasping experiments in [3]. Although the visual servoing method presented in [3] works well, we sometimes observed the problem of having a *worse* gaze configuration for the tracker. Indeed, the quality of the gaze configuration (with regard to the tracker performance) is independent of our control strategy at that time which simply keeps the objects of interest in the image (e.g. by centering them). Therefore, we sought a solution to this - a gaze control that will explicitly improve the vision algorithm.

The problem we stated fits into the term *active perception* as defined in [5] - meaning to change a sensor's state parameters in order to improve the sensing strategy, with *active vision* [6] referring to the use of vision sensors. The method here is a subset of active vision where the camera pose is changed online (without a pre-computed plan). In this particular paper, we are trying to improve visual pose estimation (sensing strategy). Because active vision is a broad field receiving a lot of interest, several works had already been compiled into a survey [7]. Although active pose estimation is not categorized in [7], it is very similar to active localization/navigation where several examples are given. However, since the methods vary widely, it is more useful to concentrate on the methodology rather than application area to position this paper with respect to the state-of-the-art. Here, we propose to create a task function based on optimality metrics applied to the Fisher Information Matrix (FIM). We then use this in a multi-task optimization framework. The first part of optimizing some information measure (not limited to the FIM) is a fairly common approach in active vision [7]. The main novelty here is in incorporating it with other tasks. We provide here a clear perspective on integration with multi-task frameworks such as [8], as we did in [3]. This paper is work towards defining active vision tasks within the framework of [8]. Furthermore, to the best of our knowledge, there hasn't been a satisfactory solution to singularities of the three image point visual servoing problem. Since the solution we present here is made possible by incorporating other tasks with active vision, the problem is a nice benchmark example of the method. Singularities are a good benchmark because these are degenerate configurations where information is lost, so then an active vision task must necessarily be able to avoid these. One can also note a clear parallel between pioneering work on singularity avoidance by defining a *manipulability* criteria [9] and recent work on active vision for the structure from motion problem [10]. Another way to describe the active vision approach in this paper is creating an *observability* criteria relating to pose estimation. This distinction is important in closely related papers such as [11] which designs a visual servoing controller with singularity avoidance for the robot mechanism, building on [9]. Whereas here, we are concerned with the singularities from vision. In fact, there is no mechanism singularity since we are simulating an ideal free-floating camera.

The rest of the paper is structured as follows. Sec. II gives a brief review of image formation and visual servoing in an optimization framework. Sec. III shows how we formulated an active vision cost function that is designed to avoid degenerate configurations for pose estimation. Sec. IV describes the three image point visual servoing problems and motivates our use of it as an example. Details are then given on how our method is applied to it. Sec. V then presents and

[1]Inria Rennes - Bretagne Atlantique, Lagadic group, Campus de Beaulieu, 35042 Rennes, France.

discusses the simulation results. Finally, Sec. VI concludes and outlines some possible future works.

II. Review of visual servoing

We start this section with formally defining image formation. Doing this, we can later show the clear link between visual servoing and visual pose estimation. Without loss of generality, it is useful to define objects as a collection of n 3D points, ${}^{\mathrm{o}}\mathbf{p}_{\mathrm{n}} \in \mathbb{R}^{3\mathrm{n}}$. These are then projected into corresponding 2D image points, ${}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}} \in \mathbb{R}^{2\mathrm{n}}$, as a function of the camera pose relative to the object ${}^{\mathrm{c}}\mathbf{T}_{\mathrm{o}}$ such that a simple image formation model is described by:

$$ {}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}} = f({}^{\mathrm{c}}\mathbf{T}_{\mathrm{o}}, {}^{\mathrm{o}}\mathbf{p}_{\mathrm{n}}). \tag{1} $$

For example, a commonly used model of image formation is perspective projection. For simplicity, taking a single point, (1) becomes:

$$ {}^{\mathrm{i}}\mathbf{p} = \left(\frac{1}{{}^{\mathrm{c}}Z}\right) \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \left({}^{\mathrm{c}}\mathbf{R}_{\mathrm{o}}\, {}^{\mathrm{o}}\mathbf{p} + {}^{\mathrm{c}}\mathbf{t}_{\mathrm{o}}\right), \tag{2} $$

where ${}^{\mathrm{c}}\mathbf{R}_{\mathrm{o}}$ and ${}^{\mathrm{c}}\mathbf{t}_{\mathrm{o}}$ are the rotation matrix and translation vector of ${}^{\mathrm{c}}\mathbf{T}_{\mathrm{o}}$ while ${}^{\mathrm{c}}Z$ is the depth component of $({}^{\mathrm{c}}\mathbf{R}_{\mathrm{o}}\, {}^{\mathrm{o}}\mathbf{p} + {}^{\mathrm{c}}\mathbf{t}_{\mathrm{o}})$.

A first-order motion model can be obtained by taking the time derivative of (1). For example, [1] shows how this is done such that we can relate the camera spatial velocity ${}^{\mathrm{c}}\mathbf{v}$ to the motion of the image points:

$$ {}^{\mathrm{i}}\dot{\mathbf{p}}_{\mathrm{n}} = \mathbf{L}_{\mathrm{pn}}\, {}^{\mathrm{c}}\mathbf{v}, \tag{3} $$

where $\mathbf{L}_{\mathrm{pn}} \in \mathbb{R}^{2\mathrm{n}\times 6}$ is known as the *stacked* interaction matrix or image Jacobian in visual servoing literature [1]. For example, for a single point ${}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}} = [x\ y]^\top$ with depth Z the interaction matrix is:

$$ \mathbf{L}_{\mathrm{pn}} = \begin{bmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix}. \tag{4} $$

The model, (2), and its linearization, (3), have proven to be sufficient given an estimate of the depth. In fact, different depth approximations have shown to be effective [1].

The model in (3) can be used to control the camera velocity by creating a task function. This is often designed such that the set of image feature points ${}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}}$ converge exponentially to a corresponding set of references ${}^{\mathrm{i}}\mathbf{p}^*_{\mathrm{n}}$. Formally, the task error is $\mathbf{e} = {}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}} - {}^{\mathrm{i}}\mathbf{p}^*_{\mathrm{n}}$ and it is designed to have an exponential convergence such that $\dot{\mathbf{e}} = -\lambda\mathbf{e}$, where λ is a gain corresponding to convergence rate. We can then use this visual servoing task function in an optimization framework [3]. For example:

$$ \begin{aligned} {}^{\mathrm{c}}\mathbf{v} = \underset{{}^{\mathrm{c}}\mathbf{v}}{\operatorname{argmin}} \quad & \left\|{}^{\mathrm{c}}\mathbf{v}\right\|^2 \\ \text{subject to} \quad & \mathbf{L}_{\mathbf{e}}\, {}^{\mathrm{c}}\mathbf{v} = -\lambda\mathbf{e}, \end{aligned} \tag{5} $$

is a standard Quadratic Programming (QP) problem with efficient solutions as described in [12]. In (5), the task function is a constraint defining a solution space. In cases where the solution is not unique, the one minimizing the camera velocity norm is chosen. This construction is similar to applying the Moore-Penrose pseudoinverse [13]. However, here we will be using a *looser* unconstrained form. First, let us define the visual servoing task function as the quadratic objective function:

$$ f_{\mathrm{vs}}({}^{\mathrm{c}}\mathbf{v}) = \frac{1}{2}\left\|\mathbf{L}_{\mathbf{e}}\, {}^{\mathrm{c}}\mathbf{v} + \lambda\mathbf{e}\right\|^2. \tag{6} $$

We can then define the the unconstrained QP problem:

$$ {}^{\mathrm{c}}\mathbf{v} = \underset{{}^{\mathrm{c}}\mathbf{v}}{\operatorname{argmin}} \quad f_{\mathrm{vs}}({}^{\mathrm{c}}\mathbf{v}) + w\left\|{}^{\mathrm{c}}\mathbf{v}\right\|^2. \tag{7} $$

This can be seen as solving the least squares task function with a Tikhonov regularization [14]. However, it may be more useful to look at (7) as relaxing the constraint in (5) by adding slack variables [13], [14]. For our discussion here, the important point is that (5) is preferred for this simple case since the single task function is strictly prioritized over the regularization term $\left\|{}^{\mathrm{c}}\mathbf{v}\right\|^2$, whereas (7) relies on the weight w to determine priority. Because of non-strict prioritization, there is also a small error on the higher priority task. The equivalence between (7) and (5) is approached as w approaches zero. More generally, for multiple tasks, the ratio of the weights is considered. The advantage of (7) is the simplicity in adding more tasks - there is no need to worry about over-constraining the problem. Finally, although a closed-form solution is available for (7), we will be presenting it in this form in the interest of scalability - both in incorporating more tasks as shown here and in testing other prioritization schemes which is possible future works.

III. Active vision in pose estimation

The pose estimation problem can be described by referring back to (1), where we are given a prior model of the object 3D points (relative to each other), ${}^{\mathrm{o}}\mathbf{p}_{\mathrm{n}}$. We then are able to observe/measure the projection of these points in the image ${}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}}$. Given that we know the correspondence of the image and 3D points, we want to estimate the relative pose between the camera and object, ${}^{\mathrm{c}}\mathbf{T}_{\mathrm{o}}$. So, for pose estimation, the model of (1), (2) can be seen as the measurement function. To continue, we first need to make some assumptions. First, the image measurements ${}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}}$ can be modeled as a Gaussian distribution whose mean is the correct image point projection. Secondly, the covariance of the normal distribution is constant and known. With these assumptions, the Fisher Information Matrix, $\mathbf{F}$, for the model used in (2) is simply:

$$ \mathbf{F} = \mathbf{L}_{\mathrm{pn}}^\top \mathbf{S}_{\mathrm{n}} \mathbf{L}_{\mathrm{pn}} \tag{8} $$

where $\mathbf{S}_{\mathrm{n}} \in \mathbb{R}^{2\mathrm{n}\times 2\mathrm{n}}$ is the inverse of the measurement noise covariance matrix. The Fisher Information Matrix is well-used in experimental design as a metric to improve the estimated quantity. In this regard, there have been several optimality metrics. These seek to optimize different quantities related to $\mathbf{F}$. Some of these are:

- E-optimality: maximize the smallest eigenvalue of $\mathbf{F}$
- T-optimality: maximize the trace of $\mathbf{F}$
- D-optimality: maximize the determinant of $\mathbf{F}$

These are all related since the trace is the sum of the eigenvalues and the determinant the product. Although E-optimality is arguably the best of these [15], there are issues

when the associated eigenvectors are reordered. A potential solution was explored in [15]. The main concern for both T and D-optimality is that the larger eigenvalues may dominate the metric. However, the effectiveness of D-optimality has been previously demonstrated in active vision in [10] and in the manipulability criteria [9]. So T or D-optimality can be sufficient but we will need to keep in mind the drawback. Here, we will only be showing results using T-optimality which is the simplest among the three. This also completes our claim that either of the three metrics can be used in active vision. However, we will continue the next explanations to be general enough to use either of the three.

The optimality metric can be defined as:

$$\mathbf{m} = m(\mathbf{F}), \tag{9}$$

where $m(\cdot)$ is the operator of the chosen metric. We will later define the task error from this. Now we can linearize by differentiating (9) as follows:

$$\dot{\mathbf{m}} = \frac{\partial m(\mathbf{F})}{\partial {}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}}} \, {}^{\mathrm{i}}\dot{\mathbf{p}}_{\mathrm{n}}. \tag{10}$$

Since ${}^{\mathrm{i}}\dot{\mathbf{p}}_{\mathrm{n}}$ was already defined in (3), we just need to define:

$$\mathbf{L}_{\mathbf{F}} = \frac{\partial m(\mathbf{F})}{\partial {}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}}}. \tag{11}$$

Since $\mathbf{F}$ is a function of $\mathbf{L}_{\mathrm{pn}}$, which is in turn a function of ${}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}}$, obtaining $\mathbf{L}_{\mathbf{F}}$ is fairly straightforward except for E-optimality. For completeness of this paper, using the trace as a metric for a single point with:

$$\mathbf{S}_{\mathrm{n}} = \begin{bmatrix} \frac{1}{{\sigma_{\mathrm{x}}}^2} & 0 \\ 0 & \frac{1}{{\sigma_{\mathrm{y}}}^2} \end{bmatrix}, \tag{12}$$

the task metric is:

$$\mathbf{m} = \mathrm{tr}(\mathbf{F}) = \frac{y^2}{{\sigma_{\mathrm{y}}}^2 Z^2} + \frac{x^2}{{\sigma_{\mathrm{x}}}^2 Z^2} + \frac{1}{{\sigma_{\mathrm{y}}}^2 Z^2} + \frac{1}{{\sigma_{\mathrm{x}}}^2 Z^2} + \frac{\left(y^2+1\right)^2}{{\sigma_{\mathrm{y}}}^2} + \frac{x^2\,y^2}{{\sigma_{\mathrm{y}}}^2} + \frac{x^2\,y^2}{{\sigma_{\mathrm{x}}}^2} + \frac{y^2}{{\sigma_{\mathrm{x}}}^2} + \frac{\left(-x^2-1\right)^2}{{\sigma_{\mathrm{x}}}^2} + \frac{x^2}{{\sigma_{\mathrm{y}}}^2}. \tag{13}$$

The corresponding Jacobian is then:

$$\mathbf{L}_{\mathbf{F}} = \begin{bmatrix} \frac{2\,x}{{\sigma_{\mathrm{x}}}^2 Z^2} + \frac{2\,x\,y^2}{{\sigma_{\mathrm{y}}}^2} + \frac{2\,x\,y^2}{{\sigma_{\mathrm{x}}}^2} - \frac{4\,x\left(-x^2-1\right)}{{\sigma_{\mathrm{x}}}^2} + \frac{2\,x}{{\sigma_{\mathrm{y}}}^2} \\ \frac{2\,y}{{\sigma_{\mathrm{y}}}^2 Z^2} + \frac{4\,y\left(y^2+1\right)}{{\sigma_{\mathrm{y}}}^2} + \frac{2\,x^2\,y}{{\sigma_{\mathrm{y}}}^2} + \frac{2\,x^2\,y}{{\sigma_{\mathrm{x}}}^2} + \frac{2\,y}{{\sigma_{\mathrm{x}}}^2} \end{bmatrix}^\top. \tag{14}$$

Using (10), (11) and (3), we now have a first-order model of the metric as a function of ${}^{\mathrm{c}}\mathbf{v}$ which we want to control:

$$\dot{\mathbf{m}} = \mathbf{L}_{\mathbf{F}}\mathbf{L}_{\mathrm{pn}}{}^{\mathrm{c}}\mathbf{v} \tag{15}$$

For the active vision task, we need to define an error to be minimized based on the metric, $\mathbf{m}$. Here, we use: $\mathbf{e}_{\mathbf{F}} = \frac{1}{\mathbf{m}}$ As commonly done, an exponential decrease of the error can be designed by: $\dot{\mathbf{e}}_{\mathbf{F}} = -\lambda \mathbf{e}_{\mathbf{F}}$, where again λ is a gain to tune. With this, the active vision task function can be described by the quadratic objective:

$$f_{\mathbf{F}}({}^{\mathrm{c}}\mathbf{v}) = \frac{1}{2}\left\|\mathbf{L}_{\mathbf{F}}\mathbf{L}_{\mathrm{pn}}{}^{\mathrm{c}}\mathbf{v} - \lambda \mathbf{m}\right\|^2. \tag{16}$$

Another interpretation of (16) is that it maximizes the metric by exponentially increasing it. However, using the active vision task, (16) by itself is ill-advised. As it has no real target/goal, the camera only moves to instantaneously increase the information metric. This often moves the image points out of the field of view. It is best used as a secondary task, for example, together with a visual servoing task:

$${}^{\mathrm{c}}\mathbf{v} = \underset{{}^{\mathrm{c}}\mathbf{v}}{\operatorname{argmin}} \quad w_{\mathrm{vs}} f_{\mathrm{vs}}({}^{\mathrm{c}}\mathbf{v}) + w_{\mathbf{F}} f_{\mathbf{F}}({}^{\mathrm{c}}\mathbf{v}) + w\left\|{}^{\mathrm{c}}\mathbf{v}\right\|^2. \tag{17}$$

IV. Three image point visual servoing problem

As per the section title, the example problem we tackle will have three image point features, defined by (3). The goal is to servo the three image points ${}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}}$ to the corresponding pre-defined image references ${}^{\mathrm{i}}\mathbf{p}^*_{\mathrm{n}}$. Working with only three image points is a problem found in both visual servoing [1] and pose estimation literature [16]. However, both fields suggest at least having 4 points, completely avoiding the issues coming from using only 3 points. Here, we will tackle these issues to show the efficacy of the proposed method. Summarizing, there are two main problems with using only 3 points:

- The solution is not unique - there can be four solutions
- There is a cylinder of singularities [17]

This paper is mainly concerned with the singularities which happen when the visual servoing Jacobian, $\mathbf{L}_{\mathrm{pn}}$, becomes rank deficient. To avoid these configurations, we seek to maximize the singular values of $\mathbf{L}_{\mathrm{pn}}$. This can be done by using the active vision task, (16). Maximizing the metrics based on the eigenvalues of $\mathbf{F}$ will also maximize the singular values of $\mathbf{L}_{\mathrm{pn}}$. To clarify, in the trivial case when $\mathbf{S}_{\mathrm{n}}$ is an identity matrix, then the singular values of $\mathbf{L}_{\mathrm{pn}}$ are the square roots of the eigenvalues of $\mathbf{F}$.

Although (16) is well suited to avoid the singularities, it does not handle the first issue which affects visual servoing. That is, it will converge to the closer solution which cannot be chosen. Although this can be enough for some problems, this is insufficient for our purpose here. A clear example of this will be shown in the next section. To resolve the first issue, we need a way to select the solution we would like to achieve. Usually, this is done by adding in the information of the 4th point, but doing so defeats our original purpose. So here we define a simple positioning task:

$$f_{\mathrm{pos}}({}^{\mathrm{c}}\mathbf{v}) = \frac{1}{2}\left\|\mathbf{L}_{\mathrm{pos}}{}^{\mathrm{c}}\mathbf{v} - \lambda \mathbf{e}_{\mathrm{pos}}\right\|^2, \tag{18}$$

which is developed similarly to the previous tasks. The error, $\mathbf{e}_{\mathrm{pos}}$, is the translation between the current camera position and a desired final position. The Jacobian $\mathbf{L}_{\mathrm{pos}}$ is simply a selection matrix for the translational portion of the velocity: $\mathbf{L}_{\mathrm{pos}} = [\mathbf{I}_3 \ \mathbf{0}]$. And as before, λ is simply a gain. (18) is a task that allows to move the camera toward a selected solution. It will work well enough as long as the solutions are sufficiently far and the desired goal is close enough to the desired position. For our purpose here, we need this position to be close to our desired visual servoing solution and far

from the other possible solutions. Note that the rotations are unconstrained and the goal does not need to be precisely where the final camera pose is. That is, (18) should not be the main task for controlling the camera, it should be (6). So with this, we can define the complete method as:

$$
\begin{aligned}
{}^{\mathrm{c}}\mathbf{v} = \underset{{}^{\mathrm{c}}\mathbf{v}}{\operatorname{argmin}} \quad & w_{\mathrm{vs}} f_{\mathrm{vs}}({}^{\mathrm{c}}\mathbf{v}) + w_{\mathbf{F}} f_{\mathbf{F}}({}^{\mathrm{c}}\mathbf{v}) \\
& + w_{\mathrm{pos}} f_{\mathrm{pos}}({}^{\mathrm{c}}\mathbf{v}) + w \left\| {}^{\mathrm{c}}\mathbf{v} \right\|^2 .
\end{aligned} \tag{19}
$$

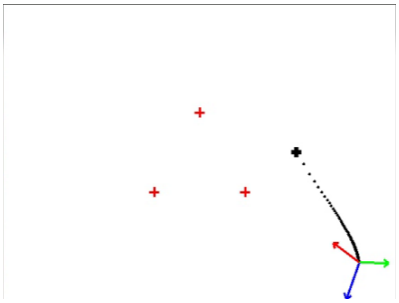
Since there are now several tasks, the issues of weighting and task prioritization will be discussed in detail in the next section.

V. Simulation Results

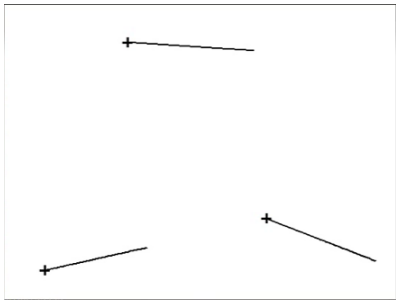
The simulations shown here are done in the software framework of the Visual Servoing Platform - ViSP [18], with an interface to qpOASES [19] for solving the quadratic optimization problems created. A simulated free-floating camera is controlled in 6 DOF through its velocity screw, ${}^{\mathrm{c}}\mathbf{v}$. This is drawn as a frame of reference with a black trajectory trail along with a black cross mark on the initial location. The reference image points, ${}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}}^{*}$ are drawn as red crosses. The current image points, ${}^{\mathrm{i}}\mathbf{p}_{\mathrm{n}}$ are drawn as black crosses with a black trajectory trail. The depth estimation is set to the current depth. As motivated in the last section, the problem has only 3 features points.

To begin, we will show the result of using (7) only. The weights are tuned to give f_{vs} more importance than regularization. The purpose here is to first show a baseline for visual servoing. It also serves to show the first issue of multiple solutions. The results are presented in Fig. 1 and 2. Fig. 1(a) shows the camera trajectory taken and the pose to which the camera converged to. Meanwhile, Fig. 1(b) shows how the features converged to the given references in image space which is further supported by the plots of Fig. 2. Although this is a valid solution for the optimization problem of (7), it is not possible to choose which solution to converge to. Furthermore, the issue of the pose estimation singularities was completely avoided by chance - the trajectory to the solution does not cross the singularity cylinder. Note that Fig. 1(a) was taken from a *top view* such that the singularity cylinder is the circle that circumscribes the three points. The next camera trajectory plots have the exact same viewpoint.

Next, we show a result of forcibly crossing the singularity cylinder without active vision. This is done by using (19) without the active vision task function and increasing the weight of the position task. Here, the reference for f_{pos} was chosen to be across the singularity cylinder. This shows that the position task works, but more importantly it serves as a comparison for when the active vision task function is added later on. Fig. 3 shows the trajectory of the camera to the intended goal. However, here we see that the singularity cylinder was crossed twice. This is because the position task is made the top priority by increasing its weight. This also means that the singularity does not disturb the control. To show one of the consequences of the singularity, we



(a) Camera trajectory in space



(b) Image point trajectory

Fig. 1. Simulation result of (7) showing the trajectory of the camera and feature points

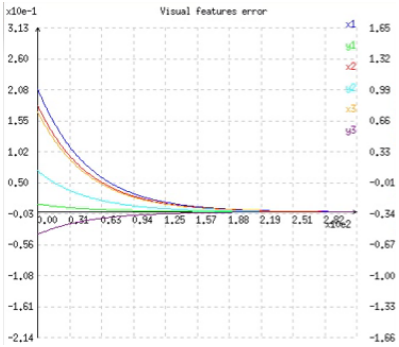


Fig. 2. Plot of the visual feature errors for Fig. 1

computed the *classical* control method in ViSP which uses a pseudoinverse on $\mathbf{L}_{\mathrm{pn}}$. Fig. 4 shows a plot of these velocities around the time of the first singularity crossing. Noting the scale of the y-axis, we see that the peaks reach about 100 m/s or rad/s. Although these values are not used in the control here, it serves as the relative reference later on. Furthermore, these high velocities are a *classical* indicator of singularities in robotics literature. Also recall that these singularities affects both visual servoing and pose estimation.

Finally, we show the results of using the full formulation of (19). The same initial conditions of the previous simulations are used. The results are presented in Figs. 5 - 8.

Fig. 5(a) shows the existence of a visual servoing solution such that the camera moves across the singularity cylinder from the initial pose. This is supported by the convergence in image space shown in Fig. 5(b) and the plots of the point feature errors in Fig. 6. Fig. 5(a) also shows that the

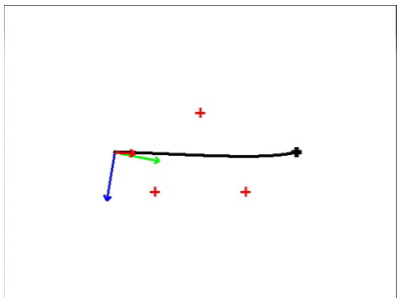


Fig. 3. Simulation which crosses the singularity cylinder

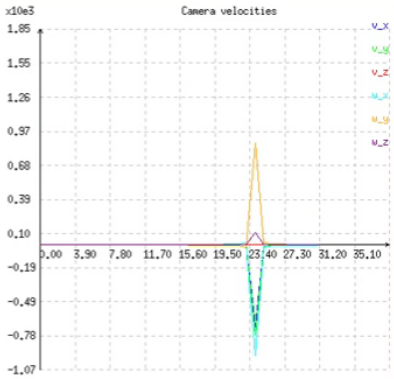
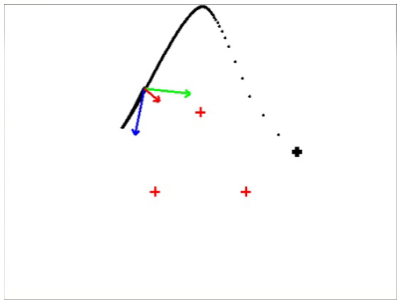
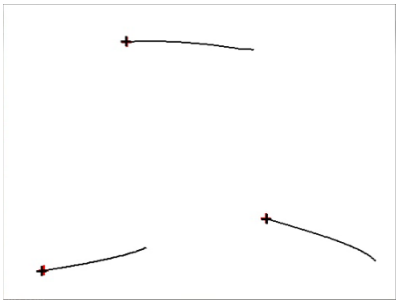


Fig. 4. Plot of the velocity from a pseudoinverse of the visual servoing jacobian for Fig. 3



(a) Camera trajectory in space



(b) Image point trajectory

Fig. 5. Simulation result of (19) showing the trajectory of the camera and feature points

camera avoids the singularity cylinder by going around it. This is the behavior we want from the active vision task designed here. Fig. 7 plots the velocities computed similar to Fig. 4. Note that there is a significant difference in the

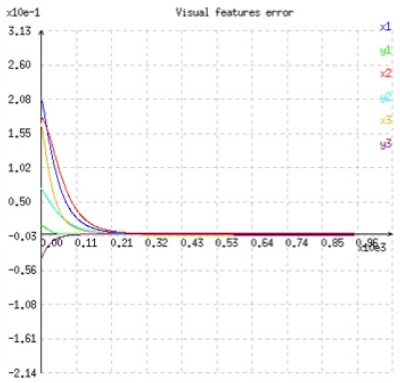


Fig. 6. Plot of the visual feature errors for Fig. 5

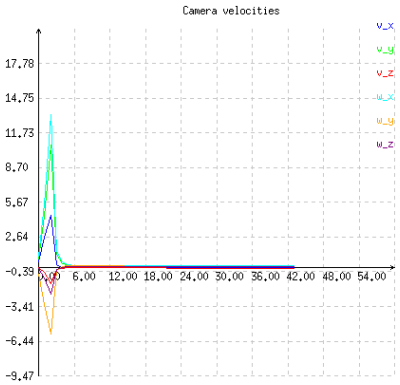


Fig. 7. Plot of the velocities for Fig. 5

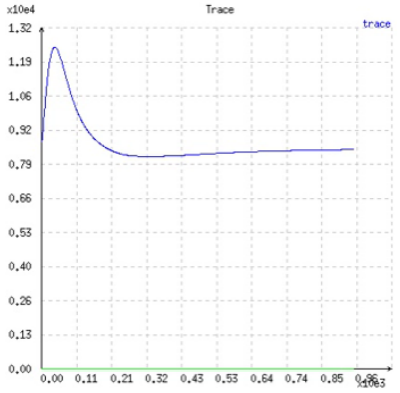


Fig. 8. Plot of the FIM trace for Fig. 5

peak values which are now around 10 m/s or rad/s. This indicates that the singularities were indeed avoided. Since T-optimality is used here, the plot of the trace is also shown in Fig. 8. Note that the avoidance behavior occurs mostly in the first portion of the servoing task from 0 to about 0.5 sec. where the trace increases. This can also be observed by the sparse trajectory trail at the starting portion of Fig. 5(a) indicating a faster velocity. However, after the initial climb, the visual servoing task dominates because it was given a higher priority/weight. At this time, the trace decreases to give way to the convergence of the point features. Lastly, towards the end when the visual servoing task converged,

there is a slow but noticeable increase of the trace as shown in Fig. 8. This can also be seen in Fig. 5(a) where the final camera pose makes it appear to backtrack on the trajectory. This can be attributed mostly to the non-strict prioritization so that task convergence is relative. Specifically, here, the small error in the visual point features is worth the increase in the trace metric.

The last point on prioritization is important to expound on. In the example here, the weights were tuned to give first priority to visual servoing, second to active vision, third to the position task and finally to regularization. Although weighting is shown to be effective here, it is non-trivial. First, for active vision, although it is treated as a secondary task to visual servoing here, one can argue that it is a higher priority than visual servoing when close to the singularities. Second, for the position task, although it is used only to push towards a given solution, the weight needs to be large enough such that closer visual servoing solutions do not dominate. Arguably it is more important when far away from the visual servoing solution. Finally, for visual servoing, although it is clearly the main task, it can be argued that the path to a solution is relatively unimportant. It is only important to converge at the end. The point is that task prioritization can vary depending on the situation. To handle this, task scheduling or adaptive weights are possible solutions. For example, one can imagine that we start with only the position task and when nearing singularities, the active vision task is activated and is the main priority. Close to the end, we can remove these other tasks and let visual servoing converge to the final solution.

Apart from prioritization, there is more work to do in defining a better active vision task. It was mentioned before to give it priority close to the singularities. A better formulation of this might be an inequality task [13] on E-optimality. Another interesting direction is clarifying the links of our approach to the field of information geometry [20]. This might lead to better task definitions.

VI. Conclusion and Future Work

In this paper, we presented our work towards active vision in a multi-task framework. For this, we described an active vision task function based on optimality metrics on the Fisher Information Matrix. This is used together with visual servoing in a multi-objective optimization framework. We then presented results on singularity avoidance in the three image point visual servoing problem.

Although the active vision task works well, there are a lot of areas to be improved. We have already briefly outlined at the end of Sec. V some possible future work on task definition, prioritization and scheduling. However, on a larger scale, it is necessary to evaluate the benefits of active vision in more complex multi-task scenarios such as [3], the final aim being to evaluate real use cases. It will also be useful to improve and refine the models here to reflect what is used in a visual tracker such as [4]. For example, taking into account the temporal nature of tracking as opposed to instantaneous pose estimation.

VII. Acknowledgement

This work was supported in part by the H2020 Comanoid project (`www.comanoid.eu`).

References

[1] F. Chaumette and S. Hutchinson, "Visual servo control, Part I: Basic approaches," *IEEE Robotics and Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.

[2] D. J. Agravante, J. Pagès, and F. Chaumette, "Visual servoing for the REEM humanoid robot's upper body," in *IEEE International Conference on Robotics and Automation*, pp. 5253–5258, May 2013.

[3] D. J. Agravante, G. Claudio, F. Spindler, and F. Chaumette, "Visual Servoing in an Optimization Framework for the Whole-Body Control of Humanoid Robots," *IEEE Robotics and Automation Letters*, vol. 2, pp. 608–615, April 2017.

[4] A. I. Comport, E. Marchand, M. Pressigout, and F. Chaumette, "Real-time markerless tracking for augmented reality: the virtual visual servoing framework," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 4, pp. 615–628, 2006.

[5] R. Bajcsy, "Active perception," *Proceedings of the IEEE*, vol. 76, pp. 966–1005, Aug 1988.

[6] J. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active vision," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 333–356, 1988.

[7] S. Chen, Y. Li, and N. M. Kwok, "Active vision in robotic systems: A survey of recent developments," *International Journal of Robotics Research*, vol. 30, no. 11, pp. 1343–1377, 2011.

[8] J. Vaillant, A. Kheddar, H. Audren, F. Keith, S. Brossette, A. Escande, K. Bouyarmane, K. Kaneko, M. Morisawa, P. Gergondet, E. Yoshida, S. Kajita, and F. Kanehiro, "Multi-contact vertical ladder climbing with an HRP-2 humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 561–580, 2016.

[9] T. Yoshikawa, "Manipulability of Robotic Mechanisms," *International Journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[10] P. Robuffo Giordano, R. Spica, and F. Chaumette, "Learning the shape of image moments for optimal 3D structure estimation," in *IEEE International Conference on Robotics and Automation*, pp. 5990–5996, May 2015.

[11] B. J. Nelson and P. K. Khosla, "Strategies for Increasing the Tracking Region of an Eye-in-Hand System by Singularity and Joint Limit Avoidance," *International Journal of Robotics Research*, vol. 14, no. 3, pp. 255–269, 1995.

[12] J. Nocedal and S. Wright, *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering, Springer New York, 2000.

[13] O. Kanoun, F. Lamiraux, and P. B. Wieber, "Kinematic Control of Redundant Manipulators: Generalizing the Task-Priority Framework to Inequality Task," *IEEE Transactions on Robotics*, vol. 27, pp. 785–792, Aug 2011.

[14] G. H. Golub, P. C. Hansen, and D. P. O'Leary, "Tikhonov Regularization and Total Least Squares," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 1, pp. 185–194, 1999.

[15] R. Spica and P. Robuffo Giordano, "Active decentralized scale estimation for bearing-based localization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5084–5091, Oct 2016.

[16] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete solution classification for the perspective-three-point problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, pp. 930–943, Aug 2003.

[17] H. Michel and P. Rives, "Singularities in the determination of the situation of a robot effector from the perspective view of 3 points," Research Report RR-1850, INRIA, 1993.

[18] É. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing: a generic software platform with a wide class of robot control skills," *IEEE Robotics and Automation Magazine*, vol. 12, no. 4, pp. 40–52, 2005.

[19] H. Ferreau, C. Kirches, A. Potschka, H. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[20] S. Amari and H. Nagaoka, *Methods of information geometry*, vol. 191. American Mathematical Soc., 2007.