Visual SLAM with an omnidirectional camera

Alejandro Rituerto
DIIS-I3A, Universidad de Zaragoza, Spain
aleritu@gmail.com

Luis Puig
DIIS-I3A, Universidad de Zaragoza, Spain
lpuig@unizar.es

J.J. Guerrero
DIIS-I3A, Universidad de Zaragoza, Spain
jguerrer@unizar.es

Abstract

In this work we integrate the Spherical Camera Model for catadioptric systems in a Visual-SLAM application. The Spherical Camera Model is a projection model that unifies central catadioptric and conventional cameras. To integrate this model into the Extended Kalman Filter-based SLAM we require to linearize the direct and the inverse projection. We have performed an initial experimentation with omnidirectional and conventional real sequences including challenging trajectories. The results confirm that the omnidirectional camera gives much better orientation accuracy improving the estimated camera trajectory.

1. Introduction

The use of omnidirectional cameras in robotics has increased in the last years. The main reason is their wide FOV. There are many omnidirectional systems, but the most used are the catadioptric systems, compound by a mirror and a camera. These systems have been used in applications such as surveillance, navigation [3] and 3D reconstruction [7]. In this work we study the application of monocular omnidirectional vision in one of the essential problems of autonomous perception and robotics: Simultaneous Localization and Mapping (SLAM).

SLAM problem appears when an autonomous vehicle does not know neither its position nor the map of its surroundings, and only have partial measurements of the environment to navigate through it. There are many SLAM approaches that use conventional cameras as sensor. Davison et al. [5] present a real-time algorithm to recover 3D trajectory of a camera moving through an unknown scene. Pupilli and Calway [9] describe a particle filtering SLAM extension which provides resilience to erratic motion. In this work we use an Extended Kalman Filter (EKF) algorithm to deal with the SLAM problem.

To obtain metric information from any vision system a projection model is required. Geyer and Daniilidis [6] propose an unified method to model any central catadioptric system. This model was extended by Barreto and Araujo [1] so that it can model central catadioptric systems and conventional cameras, and it is known as the Spherical Camera Model. This method models every system by a unit sphere and a conventional projection. We integrate this projection model into the EKF based SLAM application.

The wide FOV of the omnidirectional systems is the main advantage over the conventional vision systems. An omnidirectional system can keep tracking of image points in all directions while a conventional system can easily lose tracked points if they are not in front of the camera. In order to prove the advantages of the omnidirectional vision in SLAM we perform an initial comparison between an SLAM system using omnidirectional vision and the same application using conventional vision.

2. The Spherical Camera Model

First at all we present the projection model for the omnidirectional systems presented in [6] and extended in [1]. This model is widely used with omnidirectional vision systems.

The projection of a point in the image is explained as follows (Fig. 1). The scene point $X$ is referred to the camera coordinates centered in $O$. First, we compute the projection of the scene point $X$ to the intersection of the unit sphere and the line

---

*This work has been supported by project DPI2009-14664-C02-01. AAlso thanks to the I3A scholarship program.

†We thank J. Civera and J.M. Montiel for their basic SLAM tool and fruitful discussions.
There are two intersection points, \( x_+ \) and \( x_- \), but just one is fisically true. This fisically true point is projected to a virtual projection plane \( \pi \) trough the virtual projection centre \( C_P = (0, 0, -\xi)^T \). The new point is \( x' \). These two steps are coded in one equation:

\[
x' = h(X) = \begin{pmatrix} x \\ y \\ z + \xi \sqrt{x^2 + y^2 + z^2} \end{pmatrix}
\]

(1)

The next step transforms the virtual plane \( \pi \) in the image plane \( \pi_{IM} \) through a homographic transformation \( H_C \):

\[
x'' = H_c x'
\]

(2)

\[
H_c = K_c M_c
\]

(3)

where \( K_c \) includes the camera internal parameters and \( M_c \) includes the mirror and system parameters \([1]\). Finally, the image coordinates are computed by the next function:

\[
u = \begin{pmatrix} u \\ v \end{pmatrix} = f_u(x'') = \begin{pmatrix} \frac{x''}{\rho} \\ \frac{y''}{\rho} \end{pmatrix}
\]

(4)

The parameter of the model, \( \xi \) define the shape of the mirror we work with. For conventional cameras \( \xi = 0 \). \( \xi = 1 \) for catadioptric systems with parabolic mirror, and \( 0 < \xi < 1 \) with hyperbolic mirror.

3. Localization and Mapping

An autonomous robot must be able to navigate through unknown environments. With SLAM techniques a robot can build a map of the environment and keep track of its pose in this map using noisy measurements of the environment. In Visual SLAM these measurements are characteristic points from the images.

The most used SLAM algorithms are based on the Kalman Filter, a filter that predicts the state of linear systems. As the geometry impose nonlinear functions the Extended Kalman Filter (EKF) \([10]\) is used. The EKF linearize the non-linear relations by approximating them to its first order Taylor series, and it is divided into two parts. In the first part, Prediction, the new state of the system \( x_{k+1} \) is estimated from the previous timestep state \( x_k \).

The second part of the algorithm, Update, uses the measurements of the environment \( z \) to improve this prediction. Every state variable is represented by its mean and its covariance. In our case the system state is coded in \( x_k \).

\[
x_k = (r, q, V, \omega, x_i, y_i, z_i, \theta_i, \phi_i, \rho_i)^T
\]

(5)

where \( r_{(3x1)} \) is the system pose, \( q_{(4x1)} \) is the quaternion of the orientation and \( V_{(3x1)} \) and \( \omega_{(3x1)} \) are the linear and angular velocities. The points are coded by inverse depth \([4]\). \( (x_i, y_i, z_i) \) is the pose of the camera the first time it saw the 3D point \( i \), \( \theta_i \) and \( \phi_i \) are the angles that determinate the ray pointing to the 3D point and \( \rho_i \) is the inverse depth of the point.

3.1. The Spherical Camera Model in the EKF

The EKF algorithm needs the first derivative of the measurement equation. As explained before we use the Spherical Camera Model in the measurement equation. So, first at all we have formulated the direct projection derivatives.

- **Jacobian of the Spherical Camera Model**

\[
J = J_e H_c J_h
\]

(6)

\[
J_e = \begin{pmatrix} 1 & 0 & -x''' \\
0 & 1 & -y'''
\end{pmatrix}
\]

(7)

\[
J_h = \begin{pmatrix} 1 & 0 & 0 \\
0 & 1 & 0 \\
\frac{\xi z}{p} & \frac{\xi u}{p} & 1 + \xi z
\end{pmatrix}
\]

(8)

where \( p = \sqrt{x^2 + y^2 + z^2} \).

To initialize new features we need to estimate the covariance of the new feature from the uncertainty of the image point. Through the inverse projection of the model is possible to estimate the 3D ray where an image point lies. The estimation of the
covariance of a new feature can be done through the jacobian of the inverse projection. The inverse projection starts with the image coordinates \( u = (u, v)^T \). The point \( x'' \) is \( x'' = (u, v, 1)^T \). The equations of the inverse model are

\[
x' = H_c^{-1} x''
\]

\[
x = h(x')^{-1} = \begin{pmatrix} x' \\ y' \\ z' - \frac{\xi(x'^2 + y'^2 + z'^2)}{\xi + \chi} \end{pmatrix}
\]

where \( \chi = \sqrt{(1 - \xi^2)(x'^2 + y'^2) + z'^2} \).

So, secondly we formulate the inverse projection derivatives.

- **Jacobian of the Spherical Camera Model inverse projection**

\[
J = H_c^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}
\]

\[
\frac{\partial z}{\partial x'} = -\frac{2\xi x' (\xi x' + \chi) - (1-\xi^2)xx'\rho^2}{(\xi z'^2 + \chi)^2}
\]

\[
\frac{\partial z}{\partial y'} = -\frac{2\xi y' (\xi y' + \chi) - (1-\xi^2)yy'\rho^2}{(\xi z'^2 + \chi)^2}
\]

\[
\frac{\partial z}{\partial z'} = -\frac{2\xi z' (\xi z' + \chi) - (\xi + \frac{\xi}{\chi}) \xi \rho^2}{(\xi z'^2 + \chi)^2}
\]

where \( \chi = \sqrt{(1 - \xi^2)(x'^2 + y'^2) + z'^2} \) and \( \rho^2 = x'^2 + y'^2 + z'^2 \).

### 4. Experiments with real images

In this section we present the experiments performed with a SLAM\(^1\) application using conventional and omnidirectional images. The measurement points are codified by inverse depth parametrization [4]. The Spherical Model is used as measurement equation and combined with SIFT [8] features as measurement points. In both type of images we use a SIFT-based matching approach. We match the features in the next frame located inside an uncertainty ellipse considering a 95% of confidence.

We use two image sequences provided by The Rawseeds Project\(^2\). They were acquired with a hyper-catadioptric camera and with a conventional camera. We have calibrated the hyper-catadioptric camera [2] for better results. The ground truth is

---

\(^1\)http://www.robots.ox.ac.uk/~SSS06/Website/Practicals/SSS06.Prac2.MonocularSLAM.tar.gz

\(^2\)http://www.rawseeds.org
the absolute error for the omnidirectional camera is 0.07 radians and for the conventional camera is 0.22 radians. Fig. 4(c) shows the trajectory results for the fourth trajectory, and the Fig. 4(d) shows the orientation results. This trajectory includes many turns. However, the estimation with the omnidirectional camera performs all the turns correctly. In this case we can see clearly the scale change in the same estimation. With the conventional camera the estimated trajectory do not follow the real movement. In the case of the orientation the errors are bigger than in trajectory four. For the omnidirectional system the mean of the absolute orientation error is 0.29 radians and for the conventional camera is 0.62 radians.

Table 1 shows the mean of the absolute orientation error for every trajectory and the two vision systems. These errors have been estimated with the SLAM orientation and the odometry orientation. For every trajectory the error with the omnidirectional camera is smaller that with the conventional camera, although we use double number of frames with that system. The mean absolute error for the omnidirectional system is 0.10 radians and for the conventional system is 0.29 radians.

<table>
<thead>
<tr>
<th>Trajectory</th>
<th>Omnidirectional</th>
<th>Conventional</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.04</td>
<td>0.16</td>
</tr>
<tr>
<td>2</td>
<td>0.11</td>
<td>0.15</td>
</tr>
<tr>
<td>3</td>
<td>0.07</td>
<td>0.22</td>
</tr>
<tr>
<td>4</td>
<td>0.29</td>
<td>0.62</td>
</tr>
<tr>
<td>MAE</td>
<td>0.10</td>
<td>0.29</td>
</tr>
</tbody>
</table>

Table 1. Mean Absolute Error (MAE) of orientation in radians

5. Conclusions

We have developed a visual SLAM application based on the Extended Kalman Filter that can use catadioptric omnidirectional cameras and perspective cameras. To do this we have formulated the Spherical Camera Model Jacobians needed by the EKF. With that new application we have tested different image sequences of a hyper-catadioptric system and a perspective camera. Besides we have compared the results obtained of this testing. This comparison has shown the superiority of the omnidirectional systems in monocular visual SLAM.

References