LaTeX Author Guidelines for 3DV Proceedings

First Author
Institution1
Institution1 address
firstauthor@i1.org

Second Author
Institution2
First line of institution2 address
secondauthor@i2.org

Abstract

// TODO - insert abstract

1. Introduction

// TODO - insert introduction here

2. Related Work

// TODO - insert related work here
-Kim *et al*.-Scaramuzza -whoever build the dvs -vSLAM?

3. Dynamic Vision Sensors

// TODO - insert short information about dvs

4. Rotation Tracking

// TODO - insert section about tracking

5. Scene Reconstruction

// TODO - insert section about reconstruction
-see notes folder
We use Kalman Filters to reconstruct the grayscale image from the noisy DVS signal. For each pixel in the output image there is a Kalman Filter that keeps track of the color gradient at the pixel position. The input to the Kalman Filter is the event frequency on a line along the current pixel movement. Hereby it is assumed that the pixel's movement was constant since the last event it triggered. While this is obviously not true all the time, it is a good approximation due to the high rate of events. Note that the pixel movement is not necessarily parallel to the camera movement, e.g. if the camera is rotating around its z-axis. The pixel movement is computed with the same formulas as used in the simulationand the map lookup during tracking. Given a camera orientation, the orientation of the ray through the pixel and the camera's focal point is computed. This is then mapped to a pixel in the output image. While we use the exact position in the output image to compute the pixel movement, we do not, however, interpolate between several pixels when writing the signal to the map, but simply round the position in the output image th the closest pixel. Since we do not write the signal directly but use Kalman Filters to reduce the noise, this simplification greatly reduces the complexity of the algorithm compared to dividing the signals between several filters. The exact formulas used for the Kalman Filters can be seen in **??**. In the current state of our program, we assume that the camera's initial field of view is already known and included in the map. With this assumption we do not have to deal with a special initialization procedure but can immediately start with the standard iteration while being relatively sure that the camera movement is tracked correctly.

6. Our Work

// TODO - insert section to explain what we did here

7. Problems

// TODO - insert section about problems
-map initialization -runtime -¿ couldn't test with camera -MATLAB

8. Experiments

// TODO - insert section about experiments
-initialization -integrate whie removing shutter -integrate over short interval
-camera calibration

8.1. Camera Simulation

// TODO insert section to explain camera simulation
-use image as input -assume 360 deg view -compute camera fov for given orientation -move camera over scene -sum up differences between patches -problem: discrete -¿ use tiny steps -¿ slow

9. Results

// TODO - insert section about results

References