PROJECT IDEAS:

These are just prospective ideas that interested me. We could always implement a different technique to solve the same application or apply the algorithm for a different scenario.

1. Orientation tracking using UKF

UKF is used to predict the orientation of a rigid body. Orientation can be represented in several ways like Euler angles, Rotation vectors (axis/angle), 3x3 matrices, Quaternions etc. In most of the papers on estimating orientation, I found them to represent orientation as quaternions, which I guess is mainly for complexity reduction. These are few papers on this idea: each has a different application, like assisting in surgery, detecting orientation of camera and generating a panorama of the images captured using that camera and so on…

<https://pdfs.semanticscholar.org/3085/aa4779c04898685c1b2d50cdafa98b132d3f.pdf>

<https://github.com/yrlu/orientation_tracking-unscented_kalman_filter/blob/master/report/ese650proj2_luyiren.pdf>

<https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7027847>

1. DeepSLAM

SLAM produces a geometrical map of the environment; DL could be used to add semantic context to it, thus producing a geometrically-accurate and information rich map.

<https://arxiv.org/abs/1707.07410>

1. SLAM in AR

SLAM can be used for AR applications to recognize 3D objects & scenes, as well as to instantly track the world, and to overlay digital interactive augmentations. Check out Wikitude, KinectFusion!

<https://www.researchgate.net/publication/232638940_Simultaneous_Localization_and_Mapping_for_Augmented_Reality_PDF>

<https://www.spiedigitallibrary.org/conference-proceedings-of-spie/10666/106660S/Augmented-reality-integration-of-fused-LiDAR-and-spatial-mapping/10.1117/12.2304977.short?SSO=1>

1. PTAM --- Parallel tracking and mapping---- a particular SLAM implementation suitable for camera tracking in small scenes—one potential application is AR since there is a match.

PTAM is a monocular SLAM (Simultaneous Localization and Mapping) system useful for real-time 6-DOF camera tracking in small scenes. It maps the real world without needing to be initialized with real world markers like known natural feature targets, or odometry data. It is difficult to map input from a handheld camera as opposed to a robot because a camera will not have any odometry (input from movement sensors used to estimate the position) whereas a robot would. Additionally, neither can a handheld camera be moved at arbitrarily slow speeds. PTAM resolves this by estimating the position of a camera in a 3D environment and it maps the positions of points on objects in the space by analyzing and processing the input from the camera in real time. PTAM involves two main parts – the tracking of the camera and the mapping of the points. These are run in parallel on different threads of a multi-core processor. The tracking thread is responsible for estimating the camera pose (position and orientation) and also for rendering augmented graphics when PTAM is used for augmented reality. The mapping thread is responsible for mapping the points. The map is not updated after every frame, only on keyframes. This means that the processor has lots of time available per keyframe for calculation to make the map as rich and accurate as possible.

<https://github.com/Oxford-PTAM/PTAM-GPL>

<https://bitbucket.org/xanxys/ptam>

<https://www.doc.ic.ac.uk/~ab9515/ptam.html>

<http://www.robots.ox.ac.uk/~gk/PTAM/>