Orientation Estimation

Abhijeet Singh

***Abstract*— This report describes the application of Kalman Filter to track three dimensional orientation. We use Unscented Kalman Filter to track the orientation of a body using IMU sensor readings from gyroscope and accelerometer. The results are compared against the ground truth obtained from Vicon motion capturing system. We also generate real time panorama from the provided camera images and the filtered orientation estimates.**

I. INTRODUCTION

Orientation estimation is the process of deducing the orientation of a body with respect to a fixed world frame of reference. The orientation is typically visualized in terms of Euler angles which describe the angles made by the body axes (or the body frame of reference) with respect to some other reference frame. Such an estimation can be performed using various sensors like gyroscope, accelerometer, magnetometer etc. These sensors give some information about the dynamics of the body (like gyroscope) or its various properties (like direction of gravity using accelerometer). The task of combining all the sensor information, which contain noise and hence can be inaccurate at times, is done by algorithms like that of Kalman Filter. In this project we use a non-linear version of the Kalman Filter called the Unscented Kalman Filter, to track the orientation using the gyroscope and accelerometer measurements from an IMU. Thereafter, we generate a panorama of the environment from the perspective of the body whose orientation we estimate, using the camera images obtained at the time when the sensor readings were being recorded.

II. DESCRIPTION

The task can be broken into several parts - collecting data from sensors, converting the raw sensor data to physical dimensions, using the Unscented Kalman Filter (UKF) to combine the sensor information into orientation estimates and finally generating the panorama of the camera images.

A. Obtaining data from sensors

The orientation experiments were performed using the board shown in Figure 1 and the raw sensor values for gyroscope and accelerometer (together with camera images from body's perspective) were recorded and provided to us. The raw values of the sensor have to be converted to physical units before using them. This conversion is further discussed in the next section of Discussion.

*This work is part of the Project 2 of the ESE 650 class Learning in Robotics at University of Pennsylvania, taught by Prof. Daniel Lee

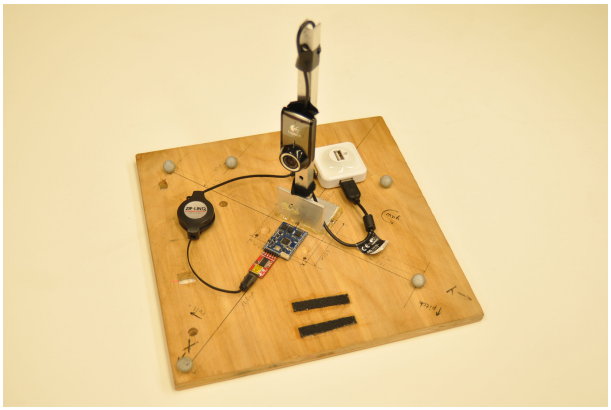

Fig. 1. Body platform with various sensors used to collect data

B. Unscented Kalman Filter (UKF)

UKF is an extension of Kalman Filter for non-linear process and measurement models. Kalman filter is an algorithm to track some state. First an estimate of the state is made which is then corrected to account for the noise during estimation using some measurements. In this project, the state is represented by a vector of orientation values together with the angular velocity components of the body with respect to the body frame of reference.

$$State = [q, \vec{\omega}]^T$$

The orientation q is represented by unit quaternion, representing a rotation from body to world frame. Since there are four elements in a quaternion, and the angular velocity $\vec{\omega}$ has three components, our state is a vector of 7 dimensions.

Kalman Filter for tracking purposes uses a process model that captures the dynamics of the system and a measurement model which estimates the measured sensor values. The difference in estimated measurement and the observed data is then used to update the state of the body.

The process model in a simple Kalman Filter captures the linear dynamics of the system, in the presence of Gaussian noise. Similarly, the measurement model assumes that the measurement estimates can be represented as a linear combination of state vector and Gaussian noise. At each estimation step, the current state is propagated through the dynamic model and the measurement model gives an estimate of the parameters being measured. The difference in the estimated and real values is then propagated through a factor of "Kalman Gain" (which is calculated from the parameters of the previous state and the measurement equation) and the state estimate is correspondingly adjusted. This

process effectively takes care of the noise in measurements by modeling it as a Gaussian and including it in the above steps.

UKF is an extension to Kalman Filter when either (or both) of the process model or the measurement model equations become non-linear. The crux of UKF is to perform the above updates on a set of "Sigma Points" that have the same statistical characteristics (like mean, covariance) as the original data. They can be calculated using Eigendecomposition or Cholesky Decomposition. These Points are transformed through a series of steps to finally obtain the estimate of the current state -

1) Project each of the sigma points (X_i) through the process model to obtain new points (Y_i). The process noise and the prior covariance can be included with the calculation of the sigma points itself, so that it is not needed here.
2) The new points (Y_i) are used to estimate the measurement vectors (Z_i) from the measurement model.
3) Compute "innovation" which describes the shift of the actual measured data from the mean of the measurement vectors (Z_i).
4) Calculate the Kalman Gain (K) and update the current state vector estimate (the mean of (Y_i)) with $K * innovation$.

The final step results in the final state estimate for the current cycle. For each new sensor data, the above process is repeated to obtain the state estimates.

C. Panorama generation

Given the camera images during the above experiment, and the orientation of the body through the UKF, we can combine the images to form a panorama. Here, we assume that all the pixels in the camera images come from a point in the sphere surrounding the camera. This greatly simplifies the panorama generation process. The basic idea is to first generate Cartesian coordinates for each pixel in the local frame and then rotate that to the world frame using the orientation obtained through the Kalman Filter. Once it is in the world frame, project the pixels back into a sphere and finally to pixel coordinates to obtain new projected image. The combination of all these new projected images gives a panorama!

III. Discussion

A. Processing raw sensor data

The raw data has to be processed to physical units before the filtering step. This way, the constants and various parameters make more sense in the UKF. The equation used to convert the data is:

$$value = \frac{(raw - bias) * vref}{1023 * sensitivity}$$

This formula is used to convert values for each axis of the measurement. The bias and sensitivity are taken from the specification of the sensors. We divide by 1023 because the sensor reading comes in 10 bit, and hence maximum value is 1023.

B. Implementing the filter

As mentioned previously, our state was a 7 dimensional vector containing 4 quaternion elements representing the rotation of the body and 3 angular velocity components, relative to the body frame. The final value of interest is just the orientation (represented in terms of quaternions here).

The process model has two parts - one related to the orientation and the other related to the angular velocities. The angular velocities are assumed to remain same (for simplicity) in the next state, and hence the process model does not alter them. The orientation is supposed to change by taking into account the previous angular velocities. The body will rotate with those angular velocities and hence it will result in a new orientation represented by the changed quaternion $q_{prev}q_{\Delta}$. The quaternion q_{Δ} is computed using previous angular velocities and change in time Δt since the previous rotation.

The measurement model also has two parts. The angular velocities are directly used from the previous process model update. This is because the angular velocities measured by the gyroscope are in the same frame as those calculated above, and hence no additional conversion is needed. The orientation part - which is measured by the accelerometer need, consists of the direction of the gravity on the body. We assume that there is no external force acting on the body and hence the accelerometer reading directly corresponds to the gravitational force. We transform the gravity vector (facing downwards in the world frame) to the local frame by using the orientation calculated in the previous process update. This gives our estimate of the acceleration reading.

The other important component in the algorithm is the noise covariance matrix for process model (Q) and measurement model (R). They respectively control the confidence in their respective estimate. These matrices are taken to be diagonal and containing only two different parts (one for orientation and the other for angular velocities), for simplicity. Hence, we have a total of 4 parameters which can be tuned to improve the performance of the filter.

Rest of the computation are directly performed using the equations mentioned in [1].

C. Generating panorama

The panorama was generated using the steps mentioned in the previous section. The pixels were mapped to spherical coordinates using the parameters horizontal field of view (Hfov) and vertical field of view (Vfov), which were taken to be 60° and 45° respectively. The conversion equations were:

$$r = 1$$

$$altitude = (nrows/2 - y) * \frac{45^\circ}{nrows}$$

$$azimuth = (x - ncols/2) * \frac{60^\circ}{ncols}$$

where we assume that the image origin is in the top left corner, x grows horizontally to the right and y grows

vertically downwards. $nrows$ is the number of pixels in the y direction and $ncols$ is the number of pixels in the x direction.

The equations to transform these spherical coordinates to Cartesian coordinates are:

$$x = r * cos(altitude) * cos(azimuth)$$

$$y = r * cos(altitude) * sin(azimuth)$$

$$z = r * sin(altitude)$$

Each of the pixels are multiplied by the rotation matrix of the nearest (in time) orientation estimate. This brings the pixels into the world frame. Thereafter, they are again converted to the spherical coordinates (using the inverse of the above equations, and taking $r = 1$). This is then projected onto a plane (by simply taking the $altitude$ and $azimuth$ as the y and x coordinates in the plane). These plane coordinates are then scaled to get the pixel coordinates which is then painted as an image. The consecutively generated images were overwritten for simplicity on the final image.

IV. Results

The results of the UKF and panorama generation have been mentioned in this section. The results are presented in two parts - first the results on the train data are presented and then those for the test data. Note that since this is not a learning method, the training data results are also a good estimate of the results - apart from the fact that the parameters in the algorithm have been tuned to do as well as possible wrt the training data.

A. Orientation angles

These plots represent the orientation angles (roll, pitch, yaw) predicted by the data. Ground truth is also mentioned in the graph in cases where it was available. In all the figures, the x-axis represents the time.

1) Train Data: Figure 2, 3, 4, 5 represent the plots when only the gyroscope data was used and integrated to calculate the orientation of the body (no filter).

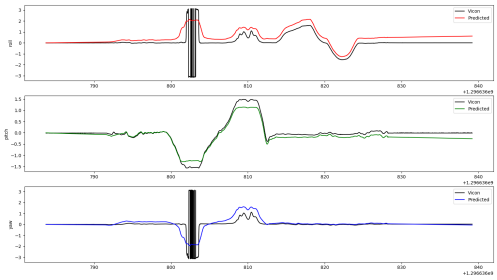

Fig. 2. Dataset 1. Angular velocity integration

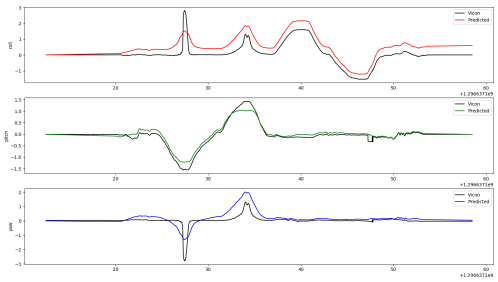

Fig. 3. Dataset 2. Angular velocity integration

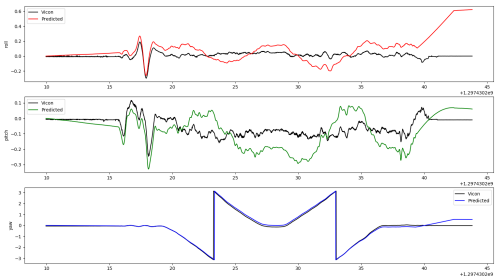

Fig. 4. Dataset 8. Angular velocity integration

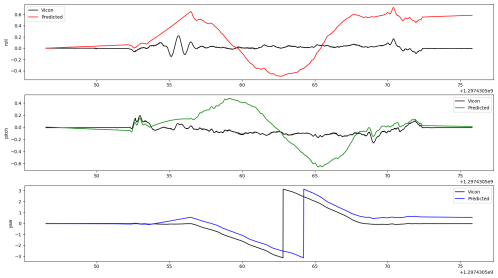

Fig. 5. Dataset 9. Angular velocity integration

We see that these predictions are able to track the orientation of the body, but drift a lot since angular velocities only measure the change in orientation and do not recover from the noise added in each step.

For the same datasets, we also plotted the graphs when only gyroscope data was used in the UKF (by taking a very very large covariance for the accelerometer measurement update so that its effect is nullified and only the gyroscope reported angular velocities are used). We also plotted the other way round - taking only the accelerometer reading and not the process update using angular velocities. The results are shown below.

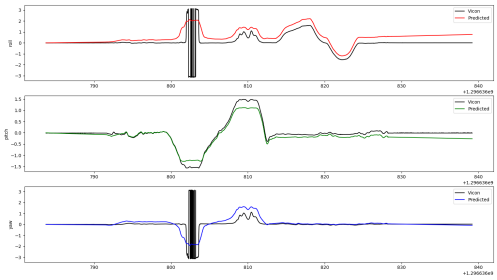

Fig. 6. Dataset 1. UKF with only gyro data.

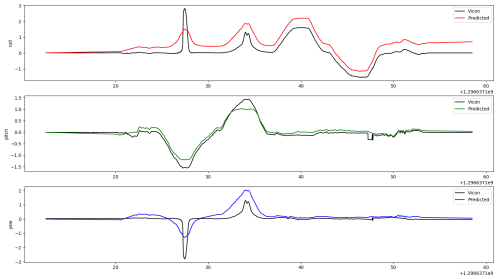

Fig. 7. Dataset 2. UKF with only gyro data.

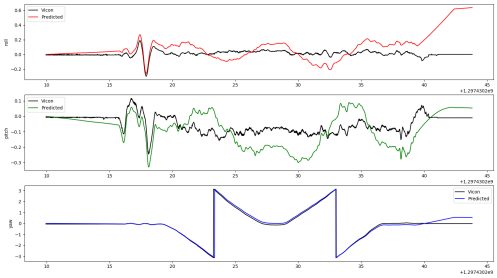

Fig. 8. Dataset 8. UKF with only gyro data.

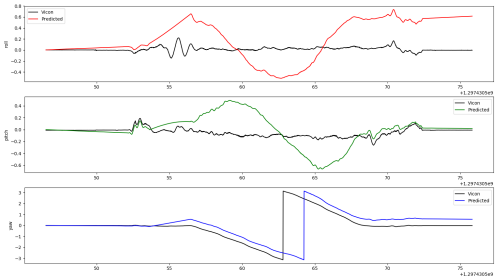

Fig. 9. Dataset 9. UKF with only gyro data.

From figures 6, 7, 8, 9 we see that the plots using UKF with only the gyroscope data are very similar to the angular velocity integration plots and suffer from the same problem as described above.

The next few figures (10, 11, 12, 13) show the plots using UKF with only the accelerometer data. There we see that there is no problem of drift and the accelerometer alone tracks the orientation of the body very well. Though it does not work as good with respect to the yaw angles (which is understandable as yaw angles are perpendicular to the direction of the gravity vector and hence the accelerometer doesn't have much information about them). Also, it fluctuates a lot in certain cases (dataset 8, 9).

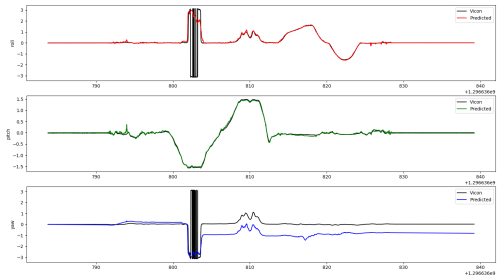

Fig. 10. Dataset 1. UKF with only accelerometer data.

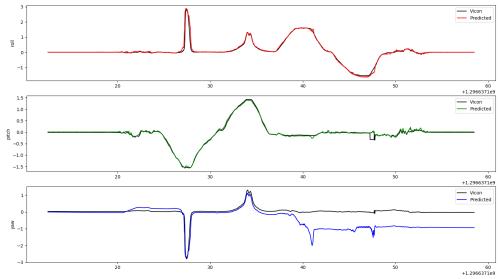

Fig. 11. Dataset 2. UKF with only accelerometer data.

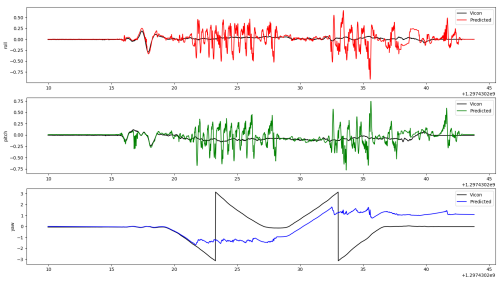

Fig. 12. Dataset 8. UKF with only accelerometer data.

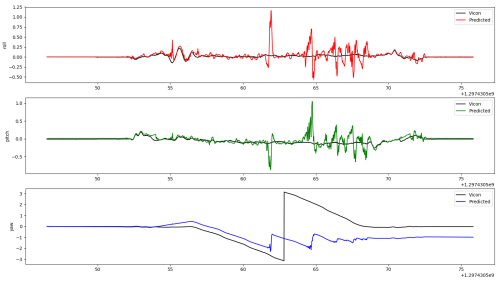

Fig. 13. Dataset 9. UKF with only accelerometer data.

The following figures show the application of both the gyroscope and accelerometer data with UKF. They seem good enough as compared to the previous plots - apart from the initial disturbance and the shift of the yaw angle prediction. We see that there is no drift as caused by using the gyroscope measurements alone and not as much fluctuation as observed by using the accelerometer alone.

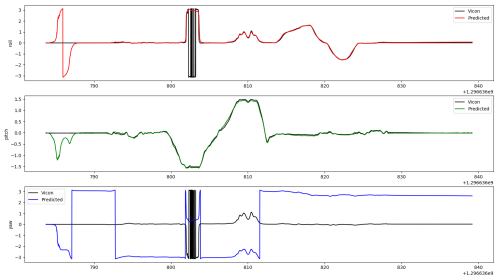

Fig. 14. Dataset 1. Full UKF result.

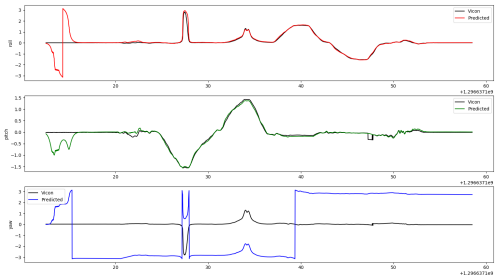

Fig. 15. Dataset 2. Full UKF result.

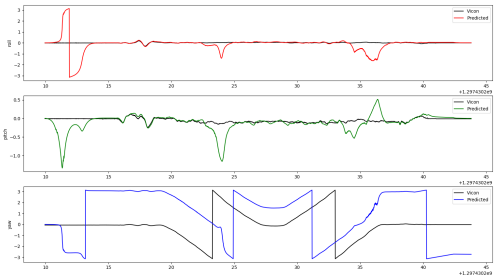

Fig. 16. Dataset 8. Full UKF result.

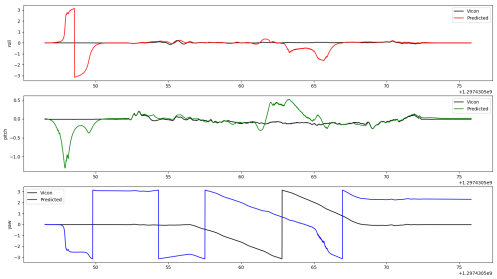

Fig. 17. Dataset 9. Full UKF result.

2) Test Data: The following are the results for the test data sets. The ground truth for dataset 10 was provided and hence shown. For the rest, we only have the predicted results.

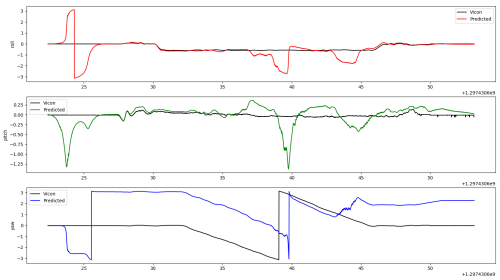

Fig. 18. Test Dataset 10. Full UKF result.

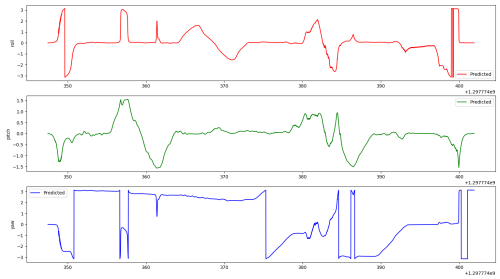

Fig. 19. Test Dataset 11. Full UKF result.

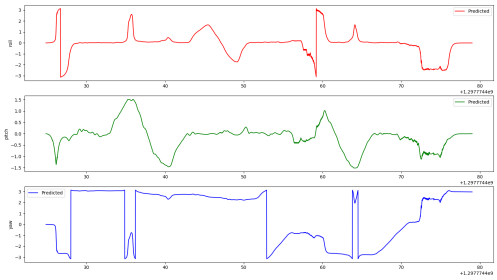

Fig. 20. Test Dataset 12. Full UKF result.

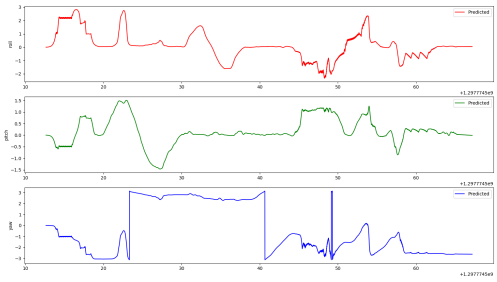

Fig. 21. Test Dataset 13. Full UKF result.

B. Panorama

The panorama results are shown in this section. The video of the panorama drawings is available at the link ***Drive***.

Some screenshots were taken during the panorama drawing process - some of the better ones are shown here.

1) Train Data: Figures 22, 23, 24, 25 show some panorama drawn from the Vicon measured orientations.

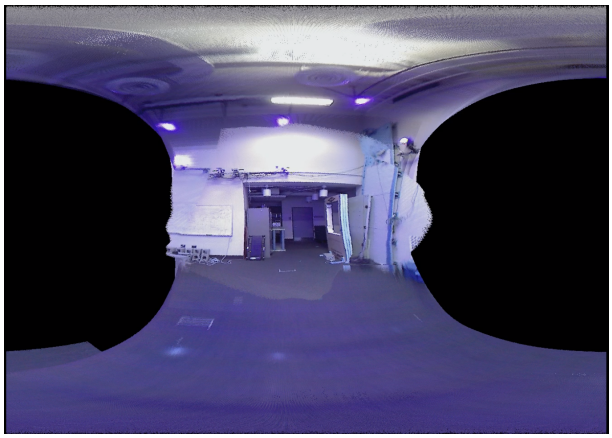

Fig. 22. Dataset 1. Panaroma from Vicon orientations.

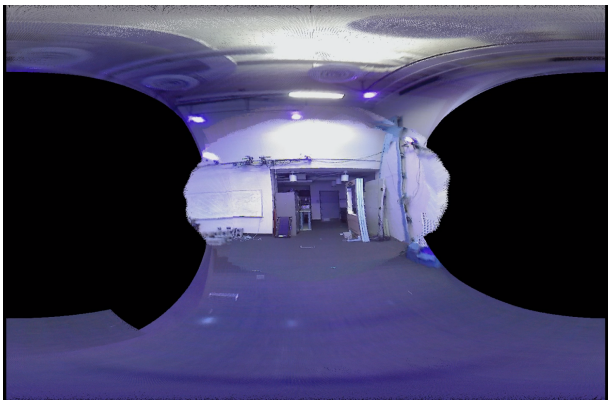

Fig. 23. Dataset 2. Panaroma from Vicon orientations.

Fig. 24. Dataset 8. Panaroma from Vicon orientations.

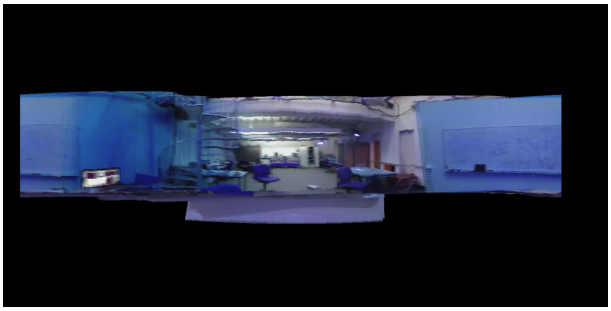

Fig. 25. Dataset 9. Panaroma from Vicon orientations.

Figures 26, 27, 28, 29 show some panorama drawn from the predicted orientations for the same dataset. They are less clear and more shifted due to the errors and fluctuations during prediction.

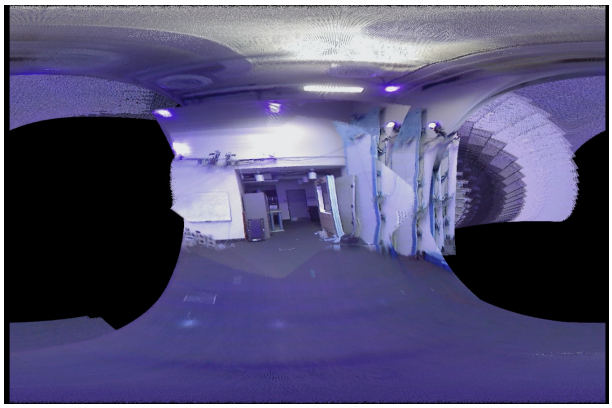

Fig. 26. Dataset 1. Panaroma from predicted orientations.

Fig. 27. Dataset 2. Panaroma from predicted orientations.

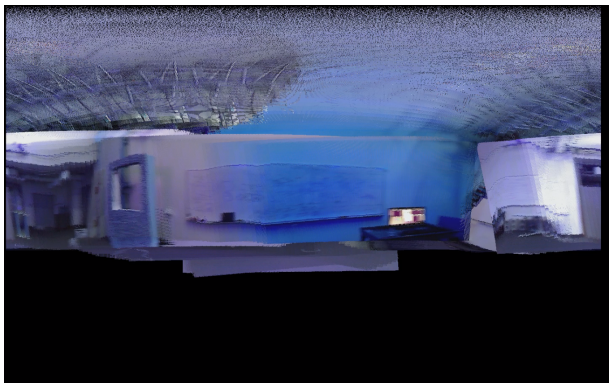

Fig. 28. Dataset 8. Panaroma from predicted orientations.

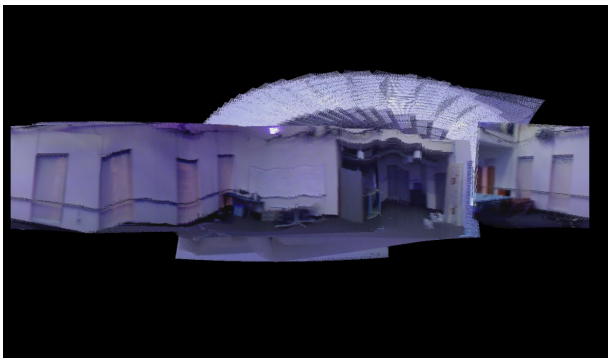

Fig. 29. Dataset 9. Panaroma from predicted orientations.

2) Test Data: The following images are taken during the panorama formation for the test datasets.

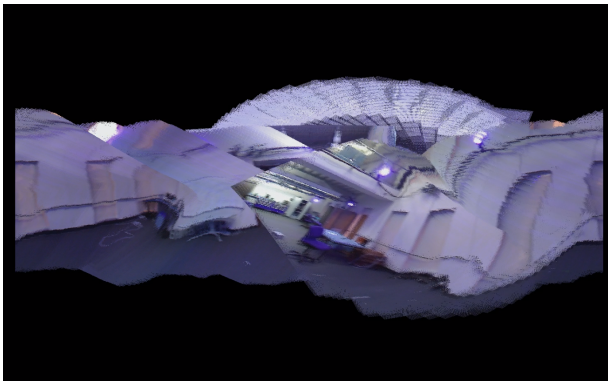

Fig. 30. Test Dataset 10. Panaroma from predicted orientations.

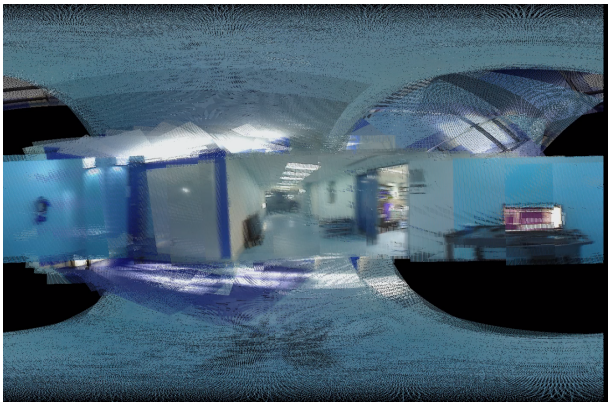

Fig. 31. Test Dataset 11. Panaroma from predicted orientations.

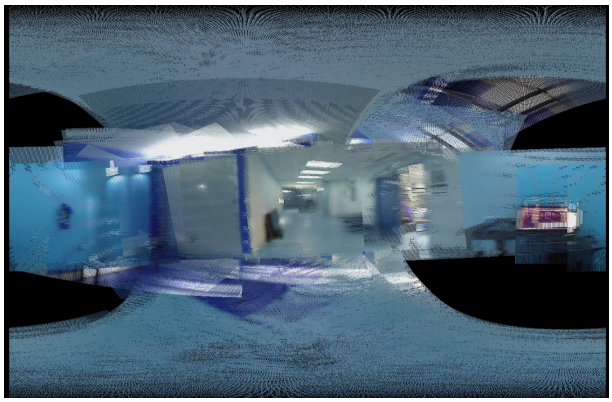

Fig. 32. Test Dataset 12. Panaroma from predicted orientations.

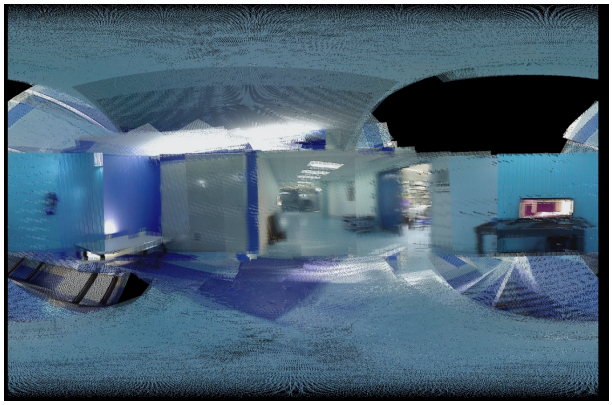

Fig. 33. Test Dataset 13. Panaroma from predicted orientations.

They are not as clear due to the errors and fluctuations during the prediction of the orientations. Though one will get a better idea by viewing the videos provided at the link mentioned above.

REFERENCES

[1] Kraft, Edgar. "A quaternion-based unscented Kalman filter for orientation tracking." Proceedings of the Sixth International Conference of Information Fusion. Vol. 1. 2003.