International Journal of the Physical Sciences Vol. 6(24), pp. 5648-5658, 16 October, 2011
Available online at http://www.academicjournals.org/IJPS
DOI: 10.5897/IJPS11.102
ISSN 1992 - 1950 ©2011 Academic Journals

Full Length Research Paper

Reducing multiplication operation and independent processing for monocular simultaneous localization and mapping (SLAM) feature state covariance matrix computation

Mohd. Yamani Idna Idris[1]*, Hamzah Arof[2], Noorzaily Mohamed Noor[1], Emran Mohd Tamil[1], Zaidi Razak[1] and Ainuddin Wahid[1]

[1]Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia.
[2]Faculty of Engineering (Electrical), University of Malaya, Kuala Lumpur, Malaysia.

Accepted 14 September, 2011

Monocular simultaneous localization and mapping (SLAM) research is a study which concentrates on how to derive position and motion estimates information from tracked features using a single camera. Before the features can be processed by standard extended Kalman filter (EKF), they have to be initialized. In the initialization process, the state covariance matrix calculation is found to be the most time consuming process. This is proven by software profiling method which is used to identify which section of program demand high processing computation. The execution time is further increased when the number of features is increased. This is due to the fact that the matrix multiplication involved in obtaining the state covariance becomes larger when more features are added. In this paper, the author proposed a new method to reduce the computation time by altering the state covariance matrix formula by reducing the multiplication operation involved. The proposed method also manipulates the conventional approach to produce multiplication process which is independent. The independency will enable future researcher to consider parallel design which would further accelerate the execution time.

Key words: Simultaneous localization and mapping (SLAM), parallel design, matrix multiplication, landmark initialization, inverse depth parameterization.

INTRODUCTION

Autonomous unmanned vehicles are predicted to be part of future urban civil and military applications [Albaker and Rahim, 2011]. One of the important areas for designing an autonomous vehicle is called simultaneous localization and mapping (SLAM). SLAM is a process where a mobile robot can build a map of an environment and concurrently use this map to compute its own location. To achieve SLAM goals, prior researchers have employed several types of range sensors such as sonar and laser sensors to be included in SLAM system. However, the range sensors are facing several disadvantages due to their high cost and data association difficulty [Durrant-Whyte and Bailey, 2006]. In the recent years, vision has become more appealing to the SLAM community. This is evidenced by a number of vision SLAM algorithms and methods as reviewed [Idris et al., 2009]. Vision sensor has been chosen due to its ability to reduce range sensors data association difficulty by providing vast amount of information. In addition, vision sensor is found to be compact, accurate, noninvasive, cheap, well under-stood and ubiquitous [Davision, 2007]. A SLAM system which use single vision sensor is called Monocular SLAM. The single camera approach is suitable for system which requires no lengthy calibration steps. Though appears attractive, Monocular SLAM is a Bearing-Only SLAM which only measures the bearing of image features. The problem with Bearing Only SLAM is that the depth information is not as straight forward as acquired by the range sensors. Depth using vision sensor

*Corresponding author. E-mail: yamani@um.edu.my, yidris@gmail.com. Tel: (603) 79676414.

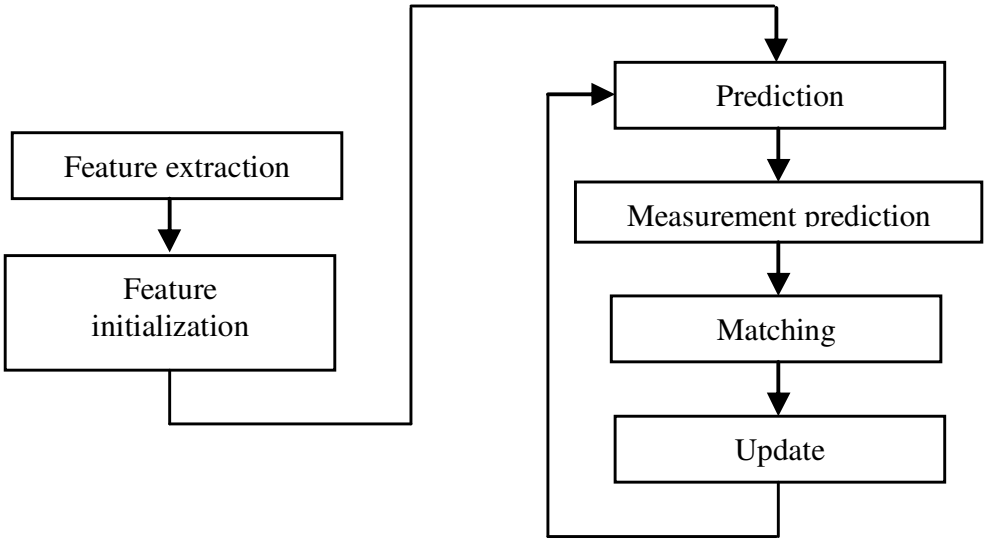

Figure 1. Monocular SLAM series of process.

can be estimated by using a feature parallax where a parallax is a measured angle of an object or captured rays viewed from two different lines of sight. In determining the depth information, the robustness of the new features initialization process has been found to be the most important problem in Bearing Only SLAM [Munguía and Grau, 2010]. This initialization problem is commonly addressed by using delayed and undelayed approach. Both approaches have their own advantages and disadvantages such that the delayed approach is able to reject weak features but has to wait until the sensor movement generates enough degree of parallax. The undelayed approach on the other hand benefits from the information about the sensor orientation from the beginning.

The downside of the undelayed approach is that the depth estimation is modeled with huge uncertainty. Moreover, its computational load grows exponentially with the number of landmarks [Munguía et al., 2010]. This is clearly shown when the state covariance matrix computation is done. State covariance matrix computation formula requires Jacobian matrix to be multiplied iteratively based on the number of features. The iterative process increases the matrix size and the multiplication process. The consequences of many multi-plication process increases the computational load and processing time. For that reason, this paper will focus on how to reduce the multiplication process by altering the original state covariance matrix formula in order to improve the robustness of the initialization process.

FEATURE INITIALIZATION EVOLUTION

A monocular SLAM is build up from series of process as shown in Figure 1. As can be seen, the features are required to be initialized before they are forwarded to the estimation process (for example, EKF). This is done to reduce uncertainty in the landmark range and to ensure the location of landmarks can be inferred from single bearing measurement. In the year 2000, Deans and Hebert [2000] proposed a Delayed initialization approach which makes use of bundle adjustment technique to compute optimal least square estimates. The bundle adjustment technique is to ensure that Kalman filter is initialized with good estimate of robot state and landmark locations. Another delayed initialization approach is proposed by Bailey and Durrant [2006] which modifies constraint initialization procedure suggested by Williams et al. [2001]. Bailey's initialization is delayed in a sense that the system will wait until base-line is sufficient to permit Gaussian initialization and become well-conditioned by using Kullback distance. Other work related to the delayed approach is presented in the paper Real-Time 3D SLAM with Wide-Angle Vision by Davison et al. [2004]. The paper shows that real time SLAM with single camera is feasible using EKF framework. They also employed particle filter to estimate the feature depth that is uncorrelated with the rest of the map. The distribution of possible depths is updated based on each new observation until the variance range is small enough to consider Gaussian estimation. The problem with such approach is that the initial particles distribution has to cover the entire possible depth values for a landmark. This would be complicated when there are many detected features or when the features are far.

Another initialization method which is commonly used by researchers is called undelayed approach. One of the earliest undelayed approaches is presented by Kwok and Dissanayake [2004]. The paper proposes that the initial representation is in the form of multiple hypotheses distributed along the direction of the bearing

measurement. For the subsequent measurements, they use sequential probability ratio test (SPRT) based on likelihoods to validate the hypotheses. In Sola et al. [2005], two drawbacks of the Delayed approach are addressed. The first drawback is the need of criteria to decide whether or not the baseline is sufficient to permit Gaussian initialization. Another drawback is that the initialization has to wait until the criteria are validated. To ensure the delay is below reasonable limits, the camera motion cannot be close to the direction of the landmark. This would be unpractical for outdoor navigation since straight trajectories are common. For that reason, Sola et al. [2005] utilize Gaussian Sum approximation that permits undelayed initialization. Since approximation representation has the tendency to be inconsistent and diverge, their work proposes federated information sharing (FIS) method to minimize the risk. In a more recent work, Monteil et al. [2006] and Civera et al. [2008] put forward a concept called inverse depth parameterization (IDP). The key idea of the concept is to produce measurement equation with high degree of linearity. Their paper discusses the drawback implied by the work done in Davison et al. [2003, 2007] which is only applicable for features that were close to camera. The drawback is caused by Euclidean XYZ feature parameterization for low parallax feature that is not well represented by the Gaussian distribution implied in the EKF. Inverse depth on the other hand is claimed to allow Gaussian distribution to cover uncertainty for both low and high parallax features. Albeit the advantage, IDP suffers from computational issue since it requires six state vector parameters (Equation 2) instead of three in Euclidean XYZ coding (Equation 1).

$$x_i = (X_i\ Y_i\ Z_i)^T \quad (1)$$

$$y_i = (x_i\ y_i\ z_i\ \theta_i\ \varnothing_i\ \rho_i)^T \quad (2)$$

where $x_i\ y_i\ z_i$ = camera optical center; θ_i = azimuth; $\varnothing_i$ = elevation; ρ_i = inverse depth.

INVERSE DEPTH PARAMETERIZATION FEATURE INITIALIZATION BACKGROUND

As discussed previously, inverse depth parameterization technique has been shown to be desirable to be implemented in a SLAM system. In this paper, the computational issue involved in inverse depth parameterization computation will be addressed. Before discussing the computational issues, background on the IDP will be presented. Overall IDP flow and formula is shown in Figure 2. This module will compute 6-D state vector (y_i) as in Equation 2 where *xi yi zi* is the camera optical center, θi is the azimuth, φi is elevation and ρi represent inverse depth. Feature initialization starts with an initial of 13 state vector (x_v) as in Equation 3 and 13×13 covariance matrix (P). The thirteen values consist of three (x,y,z) camera optical center position (r^{WC}), four (qR,qx,qy,qz) values of quaternion defining orientation (q^{WC}), three values of linear velocity relative to world frame (v^W) and another three values of angular velocity relative to camera frame (w^C). The process begins when a point feature (u,v) extracted using algorithm such as SIFT, SURF, Harris, etc is channeled to radial distortion model module. The radial distortion model will recover the ideal projective undistorted coordinates gathered by the camera. Using the "undistor a point" formula, the value of the new u and v are computed. The values of u and v will be used to find hx and hy. Together with hz, the observation of a point yi from a camera location defines a ray expressed in the camera frame as $h^C = (hx\ hy\ hz)^T$. Following that, h^C will be multiplied with the values obtained from quaternion 3D rotation process which will be used to calculate azimuth θi and elevation φi. To initialize the covariance matrix (P), similar process done in the state vector is conducted. However, the covariance matrix process requires further actions to be performed. "Undistort Jacobian" formula is utilized to acquire the $\partial h u/\partial(ud,vd)$ values which will be used to find the Jacobian J. After J has been calculated, the new covariance matrix is computed using the formula PRES=J*P*J.

$$f_v = \begin{pmatrix} r_{k+1}^{WC} \\ q_{k+1}^{WC} \\ v_{k+1}^{W} \\ \omega_{k+1}^{C} \end{pmatrix} = \begin{pmatrix} r_k^{WC} + (v_k^W + V_k^W)\Delta t \\ q_k^{WC} \times q((\omega_k^C + \Omega^C)\Delta t) \\ v_k^W + V^W \\ \omega_k^C + \Omega^C \end{pmatrix} = x_v \quad (3)$$

INVERSE DEPTH PARAMETERIZATION FEATURE INITIALIZATION SOFTWARE PROFILING

A software profiling tool has been chosen in this paper to analyze the computational issue involved in the IDP feature initialization. Software profiling is a form of dynamic programming analysis to determine which section of program demand high processing computation. The result from the profiling is shown in Figure 3. As can be seen, the calculation to determine state covariance matrix (P_RES) consume the most processing time after several iteration. The reason behind the high processing time is caused by the increasing matrix size. Covariance matrix ($P_{k|k}$) computation starts with 13×13 diagonal matrixes as discussed previously. When features are inserted, another six values (as in y_i in Equation 2) will be added into the full state vector, $x=(x_v^T\ y_1^T\ y_2^T\ \ldots..\ y_n^T)$. The insertion of more features will increase the size of the full state vector which leads to the increment of the matrix size. For instance, if one hundred features are included, a covariance matrix with size 613×613 will be used for the computation (that is, 100 (from number of features) × 6

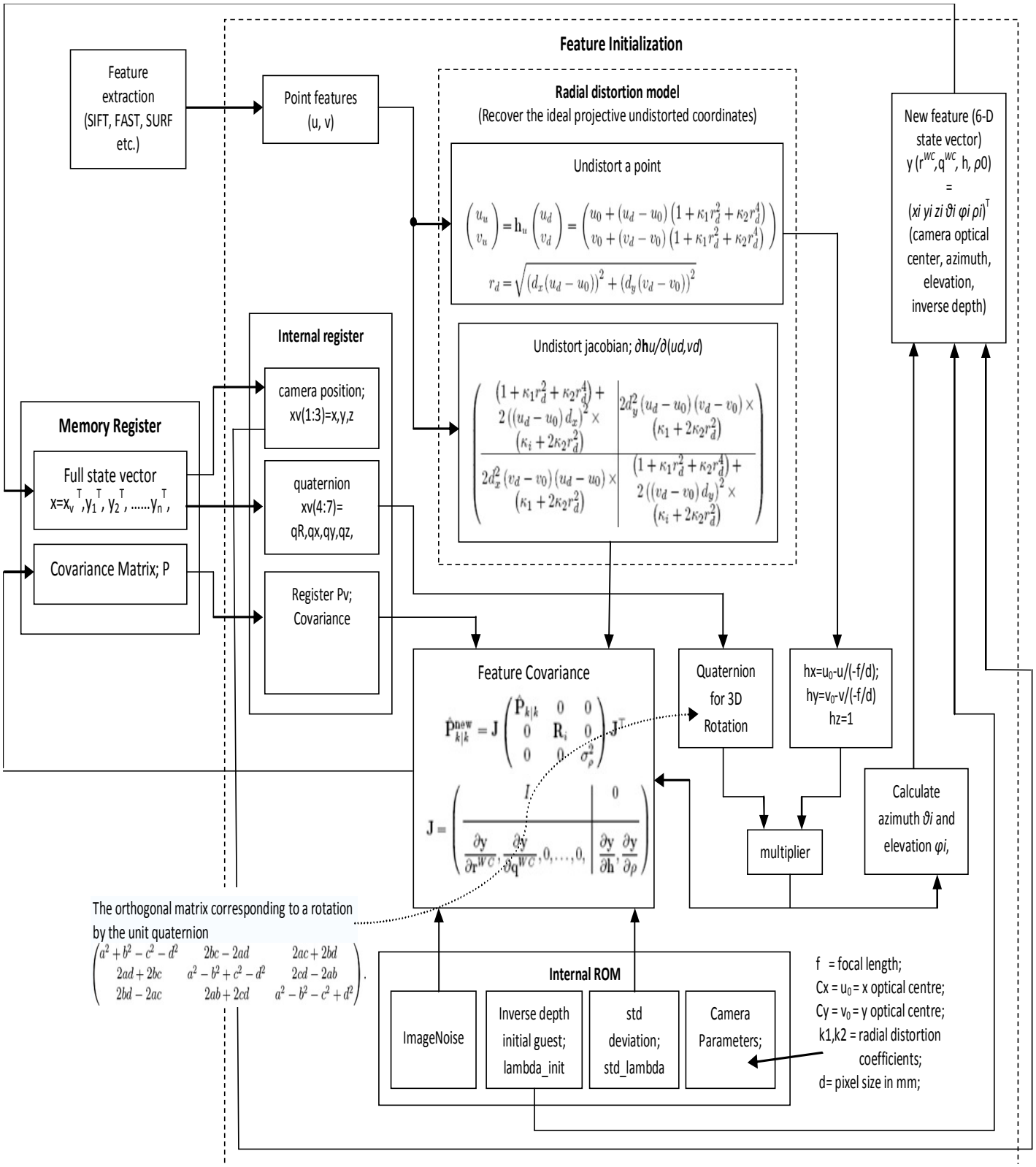

Figure 2. Inverse depth parameterization feature initialization.

(from y_i; 6-D vector) + 13 (that is, from f_v = 613). The increase in matrix size means that larger matrix multiplication has to be performed on the PRES = J*P*J which eventually increase the execution time.

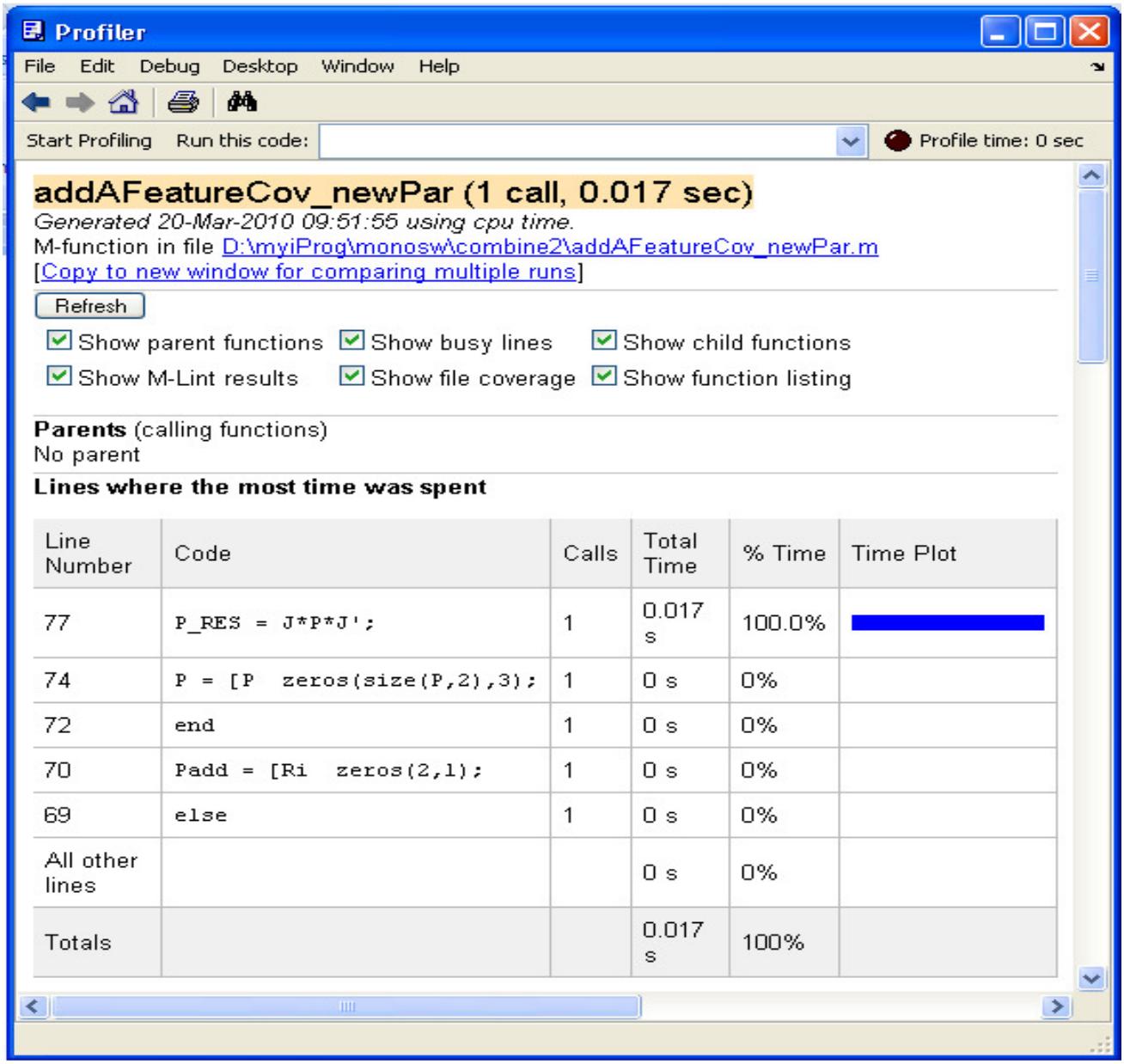

Figure 3. Inverse depth parameterization software profiling result.

INVERSE DEPTH PARAMETERIZATION AND MATRIX MULTIPLICATION

Matrix multiplication has long been studied by many researchers mainly for the reason to increase their computational speed. Classical algorithm, Winograd's algorithm and Strassen's algorithm has been discussed in our previous paper [Idris et al., 2010a, 2011]. Apart from that, several matrix multiplication implementations on FPGA are also reviewed. Among researchers utilized the FPGA to improve matrix multiplication are Prasanna and Tsai [1991], Mencer et al. [2001], Amira et al. [2001], Jang et al. [2002], Bravo et al. [2007] and Zhuo and Prasanna [2004, 2007]. We have also proposed parallel method using FPGA to improve the matrix multiplication computation in the paper. In this paper, another possible approach is investigated to find alternative solution for improving the matrix multiplication which subsequently will speed up the overall IDP process. The research concentrates on how the most time consuming state covariance matrix formula can be interpreted in a simpler way and can be calculated independently. The purpose is to reduce the computation process as well as to allow parallel processing.

Prior to the proposed solution, a discussion on how original state covariance matrix formula affects the overall

```
dydxv=[dydxv1 dydxv2 dydxv3 ……  dydxv_k]; % each dyddxv_k has constant 6x13 matrix size
dydhd=[dydhd1 dydhd2 dydhd3 ……  dydhd_k]; % each dyddxv_k has constant 6x3 matrix size

for k=1:numfeat % number of features
   J = [eye(size(P,2))        zeros(size(P,1),3); % Initial P has 13x13 matrix size & will increase
      dydxv(1:6,k*13-12:k*13) zeros(6,(size(P,2)-13))dydhd(1:6,k*3-2:k*3)];

   P = [P  zeros(size(P,2),3);
      zeros(3,size(P,2)) Padd]; % Padd has constant 3x3 matrix size

   P = J*P*J';
end
```

Figure 4. Pseudocode for calculating state covariance (*P*).

Table 1. Matrix size increment.

	1st Iteration matrix size	2nd Iteration matrix size	3rd Iteration matrix size	nth Iteration matrix size
J	19×16	25×22	31×28	19+(6)(n-1)×16+(6)(n-1)
P	16×16	22×22	28×28	16+(6)(n-1)×16+(6)(n-1)
J’	16×19	22×25	28×31	16+(6)(n-1)×19+(6)(n-1)

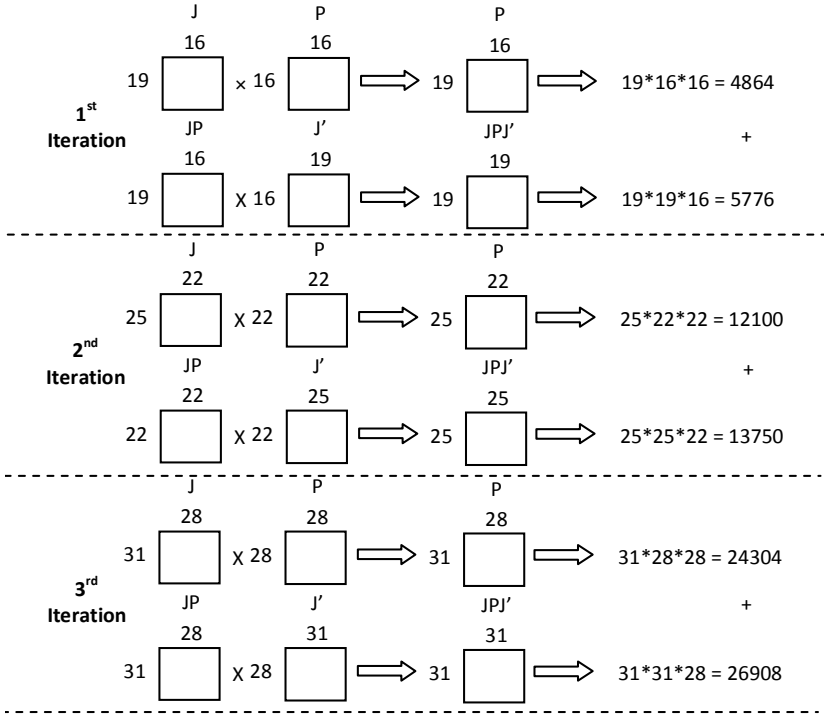

Figure 5. Number of multiplication operation needed to solve state covariance matrix using classical approach.

computation is carried out. For each feature, dydxv (that is, dy/dr^{wc}, dy/dq^{wc}, *0,….,0*) and dydhd (that is, *dy/dh, dy/dp*) of the Jacobian (*J*) are calculated. Each dydxv has a matrix size of 6×13 and dydhd has a matrix size of 6×3. A pseudocode on how the state covariance (*P*) is calculated is presented as Figure 4. The pseudocode shows that the matrix size is increasing every time a new feature is inserted. The matrix size increment is shown in Table 1 and Figure 5 illustrates how many multiplications are needed when classical matrix multiplication approach

```
dydxv=[dydxv1 dydxv2 dydxv3 ……  dydxv_k]; % each dyddxv_k has constant  6 × 13 matrix size
dydhd=[dydhd1 dydhd2 dydhd3 ……  dydhd_k]; % each dyddxv_k has constant 6x3 matrix size

for n=1:numfeat % number of features
   left=dydxv(1:6,n*13-12:n*13)*P(1:13,1:13); % dv_n X P
   right=P(1:13,1:13)*dydxv(1:6,n*13-12:n*13)'; % P X dv_n'
   rightdd=dydhd(1:6,n*3-2:n*3)*Padd*dydhd(1:6,n*3-2:n*3)'; %dd_n X Pad X dd_n'
                                                                    % Pad has constant 3x3 matrix size

   for k=1:numfeat
      dvPdvp=left*dydxv(1:6,k*13-12:k*13)'; %dv_n X P X dv_k'
      % store values in memory
      dvPdvp1=[dvPdvp1 dvPdvp];
   end

   % select diagonal from memory
   leftdv=dvPdvp1(1:6,6*numfeat*n-(6*numfeat-1)+m1:(6*numfeat*n-(6*numfeat-1))+5+m1);
   m1=m1+6;
   bottomright=leftdv+rightdd; % (dv X P X dv') + (dd X Pad X dd')

   % store values in memory
    ……
    ……

end

%Rearange P2 addresses
……
```

Figure 6. Pseudocode of the altered state covariance matrix.

is used. For three iterations, it can be seen that 4864+5776+12100+13750+24304+26908=87702 multiplication operation are needed.

PROPOSED APPROACH

Based on Figure 4 pseudocode, it can be seen that several problems could affect the processing time. The insertion of zero into the matrix *J* and *P* to suite the covariance matrix formula increase the number of operation in the program. This type of problem has been addressed by a typical approach from Zienkiewicz et al. [1971] which avoids multiplication of zero term to save time. Other problem perceived is the self matrix multiplication in P=PJP' which is expected to cause exponential growth problem every time the program iterates. The same formula is also self reliant which will cause the subsequent process to wait for prior *P* to finish computing before it can proceed.

In this paper, the aforementioned problems will be reduced. The original state covariance matrix formula is altered to avoid exponential growth problem. At the same time, a computation which is not self reliant is proposed. The purpose is to avoid waiting and to be able to process independently. Independent processing is the key to multi and parallel processing which will further speed-up the execution time. The pseudocode and diagram to illustrate the proposed design is depicted in Figures 6 and 7.

The number of multiplication involved in the proposed approach is illustrated in Figure 8. Instead of multiplying the same number, value that has been multiplied before such as *dvP* is stored in a memory. A memory look up approach is utilized to reduce multiplication operation. Table 2 shows an example of number of multiplication operation used for calculating three features. Using the proposed technique, there are (1014+1014+54+108)×3+468×3×3 = 10,782 multiplication operation needs to be performed.

PROPOSED APPROACH RESULTS AND COMPARISON

In general, the number of multiplication operation for *n* features can be compared using the pseudocode in Figure 9. The graph which compares the number of multiplication is shown in Figure 10. From the figure, it can be seen that computation of state covariance matrix using classic matrix multiplication increases exponentially. The proposed approach on the other hand, does not show large increment. This clearly shows that the proposed method is able to perform with much lesser multiplication operation. The key idea is to reuse the same calculated values which are stored in memory. Furthermore, the suggested approach is designed to avoid multiplication with zero which could be affecting the overall cycle time.

FUTURE WORK: PARALLEL PROCESSING DESIGN

As stated in the prior sections, the proposed approach reduces the multiplication operation. At the same time,

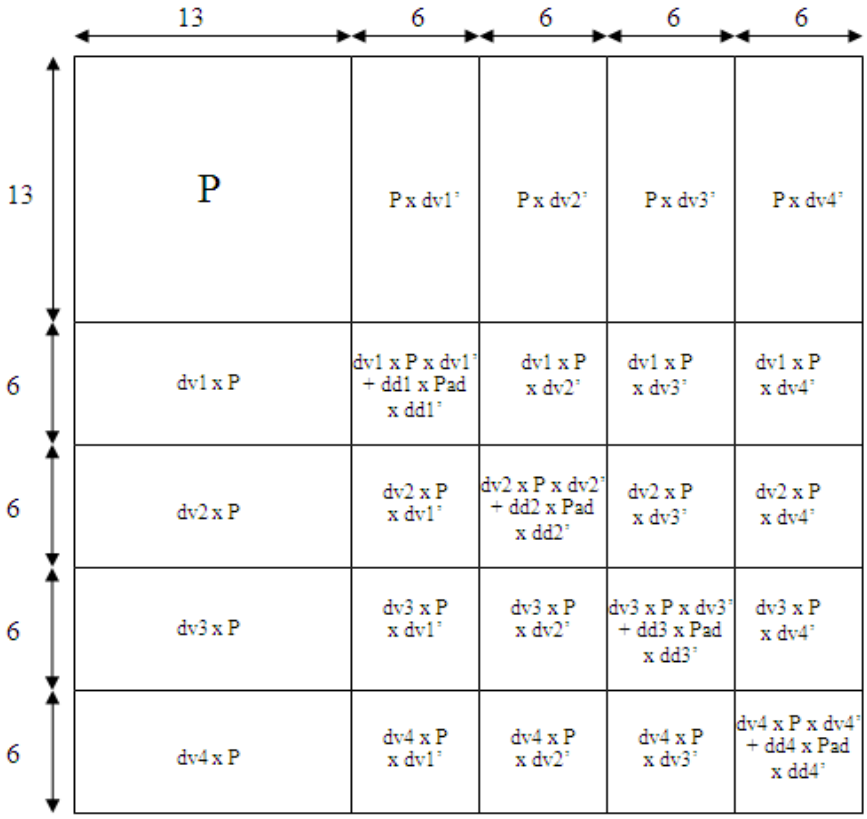

Figure 7. Altered state covariance matrix.

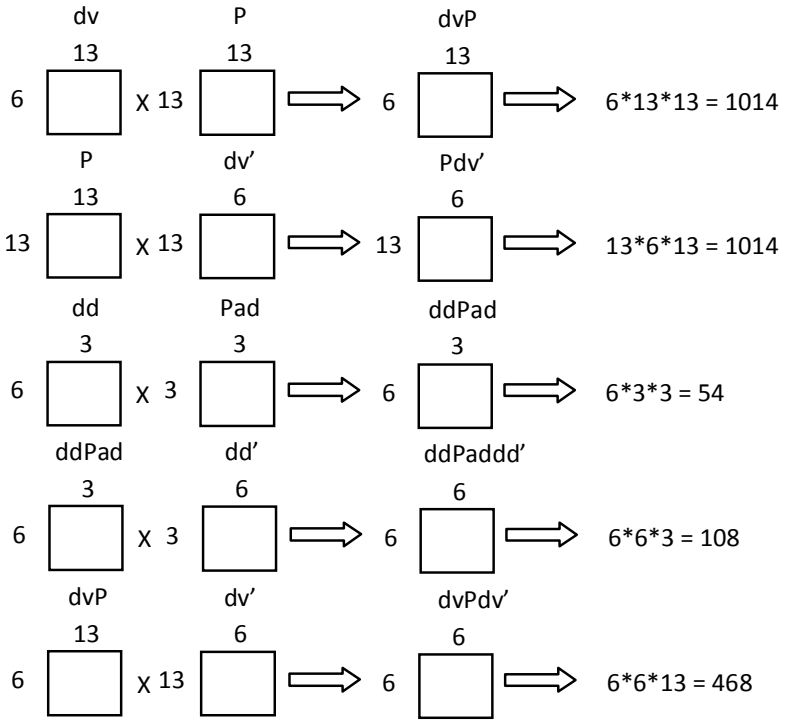

Figure 8. Number of multiplication operation needed to solve state covariance matrix using proposed approach.

Table 2. Number of multiplication operation for three features.

P	1014	1014	1014
1014	468+54+108	468	468
1014	468	468+54+108	468
1014	468	468	468+54+108

```
% number of multiplication using normal classical matrix multiplication approach
for n=1:numfeat % number of features
   normal1=((19+6*(n-1))*(16+6*(n-1)) *(16+6*(n-1))) + ((19+6*(n-1))*(19+6*(n-1))*(16+6*(n-1)));
   normal2=normal2+normal1;
   featnum=[featnum n];
   normalmult=[normalmult normal2];
end

% number of multiplication using proposed approach
for k=1:numfeat % number of features
  proposedmult1= (6*13*13+13*6*13+6*3*3+6*6*3+(6*6*13)*k)*k;
  proposedmult=[proposedmult proposedmult1];
end
```

Figure 9. Number of multiplication comparison pseudocode.

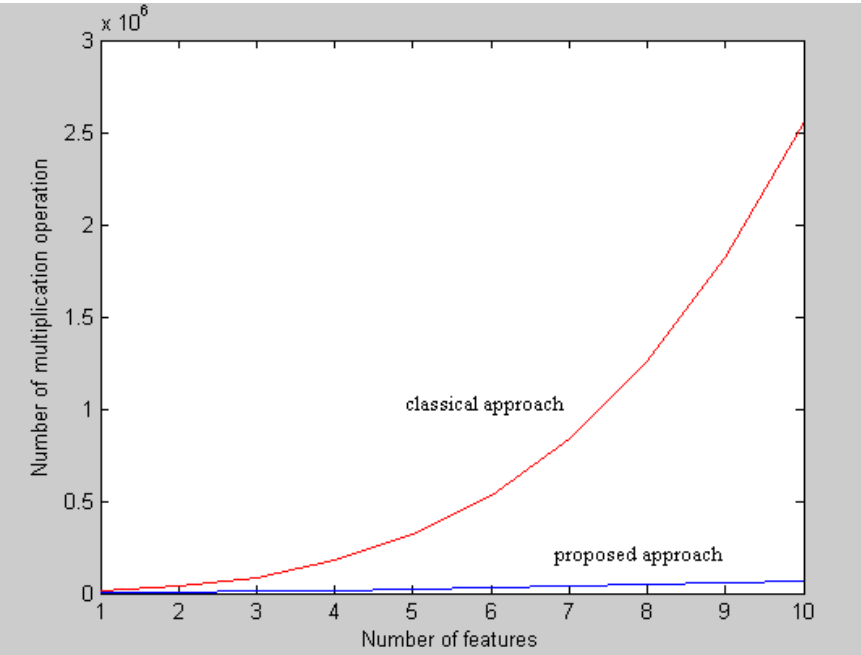

Figure 10. Number of multiplication comparison.

independent processing has been put into consideration to speed up the execution time. The purpose of independent processing is to allow each or selected features to be processed simultaneously instead of process in sequence as in normal general processor approach. To realize the independent processing, FPGA can be used to implement the parallel architecture. Figure 11 shows a parallel design that can be implemented to improve the execution time. The design separate the odd and even number features to be processed simultaneously. Though only two (odd and even) parallel processing is illustrated, more parallelism can be employed depending on the

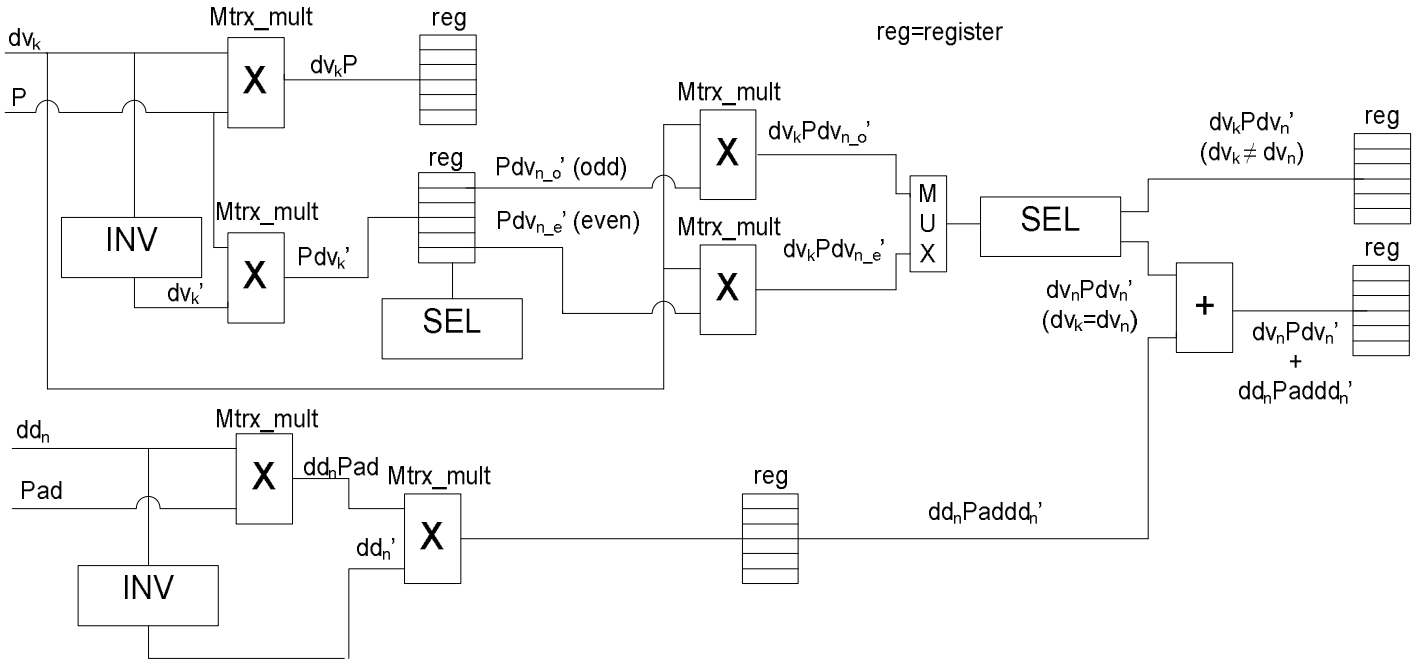

Figure 11. Parallel design.

capacity of the device chosen.

Conclusion

The purpose of this paper is to come out with an approach that will be able to speed-up the most time consuming process in the monocular SLAM feature initialization stage. First, an overview of SLAM is presented at the beginning of this paper. This is followed by feature initialization evolution review which discusses two most common initialization approaches. The two approaches are delayed and undelayed initialization approach. The undelayed inverse depth parameterization (IDP) approach is chosen since it is able to overcome delayed approach problem related to the need for a criteria to decide sufficient baseline to permit Gaussian initialization. The undelayed approach also reduces the wait before criteria are validated problem faced by the delayed approach. A software profiling tool is then utilized to search for the most time consuming process in the IDP. From the tool, it can be observed that state covariance matrix calculation contribute to the extensive processing time. For that reason, an approach to reduce such problem is proposed. The number of multiplication involved to calculate the state covariance matrix using classical matrix multiplication is analyzed. Then, the proposed method is explained and their result is presented. From the result, it can be shown that the proposed method requires much less multiplication operation compared to the original state covariance matrix formula using classical matrix approach. The reduction of the number of multiplication operation is expected to speed up the overall feature initialization process. The proposed method is also designed in such a way that enables each feature to be processed independently. This will enable future development using possible parallel architecture available in FPGA or other multiprocessing platform.

ACKNOWLEDGMENTS

This project is funded by Ministry of Science Technology and Innovation (MOSTI) under e-Science Fund (11-02-03-1047) and University of Malaya Research Grant (RG014-09ICT).

REFERENCES

Albaker BM, Rahim NA (2011). Autonomous Unmanned Aircraft Collision Avoidance System Based On Geometric Intersection Int. J. Physical Sci., 6(3): 391-401.

Amira A, Bouridane A, Milligan P (2001). Accelerating Matrix Product on Reconfigurable Hardware for Signal Processing. Field-Programmable Logic and Applications (FPL), pp. 101-111.

Bailey T (2003). Constrained Initialisation for Bearing-Only SLAM IEEE International Conference on Robotics and Automation, 2003. Proceedings. ICRA '03.

Bailey T, Durrant WH (2006). Simultaneous localization and mapping: part II. IEEE Rob. Autom. Mag., 13: 108–117.

Bravo I, Jimenez P, Mazo M, Lazaro JL, de las Heras JJ, Gardel A (2007). Different Proposals to Matrix Multiplication Based on FPGAs IEEE International Symposium on Industrial Electronics. ISIE 2007: Digital Object Identifier: 10.1109/ISIE.2007.4374862, pp. 1709-1714.

Civera J, Davison AJ, Montiel JM (2008). Inverse Depth Parametrization for Monocular SLAM IEEE Trans. Robotics, 24(5): 932-945.
Davison AJ, Cid YG, Kita N (2004). Real-Time 3D SLAM with Wide-Angle Vision 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles Instituto Superior Técnico, Lisboa.
Davison AJ, Reid ID, Molton ND, Stasse O (2007). MonoSLAM: Real-Time Single Camera SLAM. IEEE Trans. Pattern Anal. Machine Intelligence, DOI: 10.1109/TPAMI.2007.1049, 29(6):1052-1067.
Davison AJ (2003). Real-Time Simultaneous Localization and Mapping with a Single Camera. Proc. Int. Conf. Comput. Vision.
Deans M, Hebert M (2000). Experimental comparison of techniques for localization and mapping using a bearing only sensor International Conference on Experimental Robotics, Honolulu, Hawaii.
Durrant-Whyte H, Bailey T (2006). Simultaneous localization and mapping: part I. IEEE Rob. Autom. Mag., 13: 99–110.
Idris MYI, Arof H, Tamil EM, Noor NM, Razak Z (2009). Review of Feature Detection Techniques for Simultaneous Localization and Mapping and System on Chip Approach. Inf. Technol. J. DOI: 10.3923/itj.2009.250.262, 8(3): 250-262.
Idris MYI, Arof H, Tamil EM, Noor NM, Razak Z (2010). Parallel Matrix Multiplication Design for Monocular SLAM Fourth Asia International Conference on Mathematical/Analytical Modelling and Computer Simulation, AMS2010: 492-497.
Idris MYI, Arof H, Noor NM, Tamil EM, Razak Z (2011). Improving Monocular SLAM Inverse Depth Parameterization Computation Time via Software Profiling and Parallel Matrix Multiplication International J. Innov. Comput. Inform. Control ISSN, 1349-4198 7(12).
Jang JW, Choi S, Prasanna VK (2002). Area and Time Efficient Implementation of Matrix Multiplication on FPGAs, Proc. First IEEE Int. Conf. Field Programmable Technol.
Kwok NM, Dissanayake G (2004). An Efficient Multiple Hypothesis Filter for Bearing-Only SLAM Proceedings of 2004 IEEEIRSJ International Conference on Intelligent Robots and Systems September 28-October 2, 2004, Sendal, Japan.
Mencer O, Morf M, Flynn M, (2001). PAM-Blox: High Performance FPGA Design for Adaptive Computing, Proc. PDCS, 2001.
Monteil JMM, Civera J, Davison AJ (2006). Unified Inverse Depth Parametrization for Monocular SLAM. In Proceedings of Robotics: Science and Systems, August 16th-19th, 2006. Pennsylvania, USA
Munguía R, Grau (2010). A Concurrent Initialization for Bearing-Only SLAM sensors. ISSN 1424-8220.
Prasanna VK, Tsai Y (1991). On Synthesizing Optimal Family of Linear Systolic Arrays for Matrix Multiplication, IEEE Trans. Comput., 40(6): 770-774.
Williams S, Dissanayake G, Durrant-Whyte H (2001). Constrained Initialisation of the Simultaneous Localisation and Mapping Algorithm. In International Conference on Field and Service Robotics: 315–330.
Zienkiewicz OC, Taylor RL, Too JM (1971). Reduced Integration Technique In General Analysis Of Plates And Shells. Int. J. Numer. Methods Eng., DOI: 10.1002/nme.1620030211, 3: 275–290.
Zhuo L, Prasanna VK (2004). Scalable and Modular Algorithms for Floating-Point Matrix Multiplication on FPGAs, Parallel and Distributed Processing Symposium, International, 18th International Parallel and Distributed Processing Symposium (IPDPS'04), ISBN: 0-7695-2132-0, DOI 10.1109/IPDPS.2004.1303036}, 1: 92a.
Zhuo L, Prasanna (2007). V.K. Scalable and Modular Algorithms for Floating-Point Matrix Multiplication on Reconfigurable Computing Systems IEEE Transactions on Parallel and Distributed Systems, 18(4), April 2007:433-448 Digital Object Identifier 10.1109/TPDS. 2007.1001.