Generalized Preemptive RANSAC
Making Preemptive RANSAC Feasible even in Low Resources Devices

Severino Gomes-Neto[1,2] and Bruno M. Carvalho[2]
[1]*Escola Agrícola de Jundiaí, UFRN, RN 160 - Km 03, Distrito de Jundiaí, Macaíba/RN, Brazil*
[2]*Departamento de Informática e Matemática Aplicada, UFRN, Campus Universitário Lagoa Nova, Natal/RN, Brazil*

Keywords: Preemptive RANSAC, Generalized Preemptive RANSAC, Preemption Function, BRUMA, Computer Vision.

Abstract: This paper examines a generalized version of Preemptive RANSAC for visual motion estimation. The approach described employs the BRUMA function for dealing with varying block sizes and the percentages of hypotheses to be removed during the hypotheses rejection phase. The generation of a flexible number of hypotheses is also performed in order to balance the preemption scheme. Experiments were performed for both forward and side-wise motions in synthetic environment by using simulation and the ground-truth used to compare the Standard Preemptive RANSAC and its generalized version. Simulations confirmed that the quality of the results produced by the Standard Preemptive RANSAC degrade as the hardware resources used are decreased, as opposed to the results produced by the Generalized Preemptive RANSAC, with the results of the Standard Preemptive RANSAC having errors up to eleven times larger than the Generalized Preemptive RANSAC.

1 INTRODUCTION

The tasks carried in automated motion estimation are computationally expensive, resulting in a considerable amount of time dedicated to perform them. Thus, one should concentrate in achieving accurate estimations in a not prohibitive amount of time when planning a computer vision system. In other words, one needs to reach the balance between these goals in order to design a good system.

We are particularly interested in the research that focus on accelerating the methods in order to promote their feasibility in low power and/or low cost hardware setups, allowing these setups to be employed in fields such as robotics, autonomous navigation and several other applied Computer Vision (CV) tasks.

Computer Vision systems have made large use of the RANSAC algorithm (Fischler and Bolles, 1981) in order to deal with the hypothesize-and-verify problem. However, Nistér discussed that the standard RANSAC is not designed for executions under time constraints (Nistér, 2003).

The standard RANSAC first generates hypotheses from subsets of observations and then evaluates these hypotheses against all the available observations in order to determine which hypothesis is the best representative of the entire population (the closest from the ideal solution), and keeps working until it reaches a confidence threshold. This conceptual simplicity and the results achieved made RANSAC a standard in the CV and 3D Reconstruction (3DR) fields.

Nevertheless, the standard algorithm has insufficient performance in some cases. Thus, several methods were published targeting the goal of speeding up RANSAC tasks and dealing with the trade-off between quality and time by working in three major points: alternative hypothesize-and-verify methods, improvements on the hypotheses generation and improvements on the hypotheses evaluation.

The first approach has a variety of works, such as MLESAC (Torr and Zisserman, 2000), AMLESAC (Konouchine et al., 2005), KALMANSAC (Vedaldi et al., 2005), PROSAC (Chum and Matas, 2005), GASAC (Rodehorst and Hellwich, 2006) and GOODSAC (Michaelsen et al., 2006). All of them execute hypothesize-and-verify tasks but each one has its own particularities, leading to algorithms that are efficient for specific classes of applications.

Nistér's Efficient Five-Point Algorithms (Nistér, 2004) and hypotheses generation using Graphics Processor Units (GPUs) are examples of the second branch that consists of speeding up the hypotheses generation process or enhancing their quality in order to decrease the number of generation rounds. How-

Gomes-Neto S. and M. Carvalho B..
Generalized Preemptive RANSAC - Making Preemptive RANSAC Feasible even in Low Resources Devices.
DOI: 10.5220/0004737004060415
In *Proceedings of the 9th International Conference on Computer Vision Theory and Applications* (VISAPP-2014), pages 406-415
ISBN: 978-989-758-009-3

ever, it has been shown that each hypotheses generation method fits better to certain camera motions (Šegvić et al., 2007b), i.e., we do not have a single best hypotheses generation method.

The last approach focuses on accelerating the evaluation process or, alternatively, detecting a condition that avoids unnecessary exhaustive verification. This anticipation in the end task is often called preemption and it is the way we choose for improving the performance of the hypothesize-and-verify algorithm (RANSAC in our work).

The need for time constraints comes from the requirement of working in a range of time variability as narrow as possible. The time constraints assure the control over variability, but is unable to assure the preservation of estimation accuracy by itself.

We use the Nistér's Preemptive RANSAC (Nistér, 2003) (referred in the rest of the paper as P-RANSAC) as the representative of the last mentioned approach since it is the most used method after ten years of its publication.

The P-RANSAC is aware of the need of changing evaluation process in order to deliver estimations that still are in a level of accuracy compatible with vision applications. The results presented in (Nistér, 2003) made clear that it is possible to reach the balance between accuracy and speed of execution to develop useful applications. The P-RANSAC employs a function in order to identify the moment of preemption. This concept is simple and powerful, but the function defined in Nistér's work is very dependent on number of hypotheses and observations to work properly.

Afterwards, Nistér speeded up the hypotheses generation, the most costly operation in the estimation process, by using the efficient 5-point algorithm (Nistér, 2004).

Even demonstrating that the balance is possible, Nistér's works are not fit for every vision application as has been pointed out in (Raguram et al., 2008; Chum and Matas, 2008; Gomes-Neto and Carvalho, 2010). Despite the issues concerning the fact that P-RANSAC been derived from RANSAC and consequently inherits some of its limitations, the problem that is embedded in its conception is the presence of constant parameters that were chosen empirically or arbitrarily.

Empirical parameters depend on the hardware where the experiments were performed. Thus, we have a problem to reproduce the empirical parameters or to fit it to distinct hardware setups. In the case of the arbitrary parameters, one makes assumptions that should be valid for all possible setups, and, in the case of being wrong, that results in a bad problem modeling. In order to circumvent that, we employ the function BRUMA (Block Resizing for UnderManned Amount of Assays Avoidance) (Gomes-Neto and Carvalho, 2010), that performs a generalization of preemption functions and allows parameter variations. The idea is that when this preemption function is used with P-RANSAC, one can achieve the balance between accuracy and time consumption by chosen parameters that fit to particular hardware setups in what we call Generalized Preemptive RANSAC (GPR).

This work presents the results of experiments by using GPR to determine parameters that fit distinct hardware setups in order to prove the feasibility of adapting the original P-RANSAC to running in low resource devices.

The rest of this work is organized as follows. Section 2 presents the Generalized Preemptive RANSAC, while Section 3 describes the methodology employed in the experiments. Sections 4 and 5 present the experimental results and the discussion about them, respectively. Finally, Section 6 summarizes the paper, highlighting contributions and further work.

2 GENERALIZED PREEMPTIVE RANSAC

We now present the Generalized Preemptive RANSAC (GPR), a framework that uses the BRUMA function in the P-RANSAC, but also allowing the variation of an additional parameter of the preemption scheme in order to show that one fixed parameter set for all cases may not be able to balance the preemption and control the error level of the estimation task.

The first goal is to produce similar results when comparing with P-RANSAC under the same time constraints, but selecting distinct parameters to prove the aforementioned claim in a fair comparison standard. The second goal is to employ the GPR framework in restrictive conditions, where P-RANSAC's estimation quality is expected to degenerate due to lack of resources for supporting the original parameters selection, and show that GPR is more robust under the same conditions, a result obtained by just selecting the parameters in accordance with the hardware limitations.

The GPR makes use of three parameters that together can control the preemptive evaluation: the number of generated hypotheses, the size of the block of observations that rule the hypotheses elimination task and the fraction of hypotheses to remove in each rejection round.

The P-RANSAC preemption function (1) is given by

$$f(i) = \lfloor M \cdot 2^{-\lfloor \frac{i}{B} \rfloor} \rfloor, \quad (1)$$

where $\lfloor . \rfloor$ denotes the floor operator, M is the number of available hypotheses and B denotes a fixed block size (i.e. the amount of observations the algorithm must test in an evaluation round to allow hypotheses rejection). The BRUMA function can be seen as a generalization of the P-RANSAC preemption function (1) and its given by

$$f(i) = \lfloor M_i \cdot p_i^{\,-\lfloor \frac{i}{B_i} \rfloor} \rfloor, \quad (2)$$

where M_i denotes the amount of hypotheses at the i-th evaluation step, p_i is a scalar indicating the fraction of hypotheses to eliminate on a rejection round and B_i denotes the block size at the i-th evaluation step.

In both equations every time i reaches the block size the algorithm sorts the available hypotheses according their scores and discards a fraction of less promising hypotheses, i.e., the hypotheses with the lowest scores. The process is repeated until one of the stop conditions occur, either a single hypothesis remains and is the winner or the time budget is exhausted and the hypotheses with the best score is the winner.

The P-RANSAC deals with hybrid preemption scheme emphasizing breadth as defined in

$$\forall o_j, \pi_n(o_j) = \sum_{i=1}^{n} \rho(o_i, h_{i|j}), \quad (3)$$

where $h_{i|j}$ denotes that the hypothesis chosen depends on the observation, since the observations are grouped in blocks. This means that its preemption function prioritizes the coverage of hypotheses in the evaluation process. However, since P-RANSAC defines a rigid setup of parameters, it can degenerate to a pure breadth-first preemption scheme if it does not perform at least one round of elimination before exhausts the time budget.

The P-RANSAC's preemption function inspired BRUMA, with the latter allowing varying block sizes (B) and fraction of hypotheses to remove (referred as p_i or FHR, from now on). By using BRUMA instead of the original function, a system is able to adjust the parameters and fit pure depth-first, pure breadth-first or a wider number of possibilities of hybrid preemption schemes as shown in (Gomes-Neto and Carvalho, 2010) and may also avoid degeneration due to the lack of elimination rounds mentioned previously.

The usage of the BRUMA function allows us to vary two of the control parameters, with the third concerning the maximum number of hypotheses to be generated (and indirectly the portion of time dedicated to hypotheses generation).

In the generation step, P-RANSAC determines a maximum number of hypotheses to be generated. This number is fixed at 500, and, one can empirically see that it is easy to reach and pass this number when working with relaxed time budgets and/or powerful hardware. However, each CV application has a hardware setup to fit its demands and it is not rare to have modest configurations even having restricted time requirements.

Taking this into account, we hypothesized that in lower capability devices the Maximum Number of Hypotheses to Generate (MNHG) must be decreased, and confirmed it experimentally. That means that each device must have its MNHG determined by its individual capacities of generating and testing hypotheses. We will return to this point when describing experiments and their results.

In order to save time in the evaluation process, P-RANSAC defines a periodicity in which it has to perform hypotheses elimination. The period is defined by a number of observations the algorithm has already tested before elimination and is called block size in reference of the block of observations.

The determination of the block size has capital importance in the process of evaluation, since it determines the moment that the less promising hypotheses so far are eliminated. If the block size is smaller than necessary, elimination starts earlier, possibly discarding good hypotheses that just have not been tested enough to estimate their scores accordingly. On the other hand, if the block size is larger than required, a considerable number of tests may be done over useless hypotheses or the tests may not performed in the whole block, thus failing to reject any of the generated hypotheses and being the same as applying the standard RANSAC with breadth scoring.

Nistér determines experimentally that a block size of 100 observations reaches a good balance. As we mentioned before, one should be aware that balance is a matter of hardware setup. Thus, we will show experimentally that different hardware setups will have different associated optimal block sizes.

Finally, the fraction of hypotheses to be removed in each rejection round determines the amount of hypotheses that will be discarded. Since the hypotheses are scored for every evaluation step until reaching the block size, a fixed number of the hypotheses with the lowest scores are eliminated.

The P-RANSAC adopts a rejection ratio of half of the available hypotheses reducing rapidly the number of hypotheses to analyze. The concept seems promising, but we decided to investigate the impact of chang-

ing this value, since the changes in the other parameters may affect this choice.

3 METHODOLOGY

We made the option for simulating using synthetic data in order to have a groundtruth reference to compare the algorithms accordingly. The first set of simulations was planned to compare P-RANSAC and GPR in a powerful hardware as a way to compare them fairly and demonstrate that the original values of parameters used in P-RANSAC are not the unique parameter setup that works well with the balanced Preemptive RANSAC. Thus, we did not focused in surpassing P-RANSAC performance, but in achieving a set of parameters that aids us to prove such concept.

A second round of simulations were performed using a low budget hardware in order to demonstrate the slow decreasing of accuracy when using GPR as opposed to the high error results produced when using P-RANSAC.

A total of 1000 replications were simulated for each setup. We modified the simulator developed by Šegvić and used in his works for performance (Šegvić et al., 2007b) and accuracy (Šegvić et al., 2007a) evaluations. The original code[1] uses the library Boost[2] and the Oxford Active Vision Lab VW34[3].

Table 1 presents a total of six time budgets we defined to evaluate, covering a number of possible time restrictions that seems feasible for a wide range of applications. The simulation setups were planned in order to allow fair comparisons between P-RANSAC and the tuned GPR. To reach this goal we fixed some simulation parameters while the experimental parameters have their values determined and tested in each simulation setup.

Simulation parameters that remained fixed along all the experiments were the number of observations (1000), the five point hypotheses generation algorithm (Nistér, 2004), (Li and Hartley, 2006), (Šegvić et al., 2007b) and Gaussian white noise ($\sigma = 1$).

We planned to vary the number of observations, but not significant influence had been detected from such variation; thus, we fixed the number of observations to be equal to the number originally used in P-RANSAC. The option for five point algorithm was natural since it was firstly used in the P-RANSAC and Gaussian white noise was used since it is very disseminated in vision systems testing when working with synthetic data.

Table 1 presents the parameters used in the experiments which were chosen to represent several application requirements' scenarios. We used two types of synthetic motion (forward and side-wise) in order to collected the Translational Error (TError) of each algorithm in both circumstances. The motions were not combined and TError was computed separately by decomposing the winner hypothesis into the rotation matrix and translation vector, with the vector being used to determine the angular error concerning the ground truth vector.

The sets of generated hypotheses were used by both algorithms and scores were computed summing up reprojection error of every available hypothesis for a certain number of observations. In order to make sure that all the hypotheses have the same number of tests in the case of preemption by time expiration, the error for a given observation is computed for all hypotheses before moving on to another observation.

We altered the hypotheses generation step to deal with the variable MNHG and, as a consequence, we included an extra test that prevents the generation of extra hypotheses if it reaches the determined MNHG. This allowed both algorithms to add the remaining time from generation task to the one third of budget assured by default to the evaluation step.

The experiments performed to validate and test our method were the following:

1. **Validation** of the method, where we executed experiments using both algorithms with the same parameters to confirm that both algorithms produce identical results.
2. **Adjustment** of the method, where we executed experiments with adjusted parameters in GPR, but using standard maximum for hypotheses generation (500).
3. **Test Scenario 1**, where we executed experiments with different maximum number of hypotheses (5 $\times$ budget, given in μs) for each budget and with adjusted GPR parameters.
4. **Test Scenario 2**, where we executed experiments with different maximum number of hypotheses (5 $\times$ budget) for each budget and with adjusted GPR parameters in a low-budget hardware.

The validation experiments confirmed that using the same parameters in both P-RANSAC and GPR produce identical results, as it should be. The result of experiments in the other three experiments are presented in the next section.

[1]Available at http://www.zemris.fer.hr/∼ssegvic/src/evalpose.tar.gz

[2]Available at http://www.boost.org/

[3]Available at http://www.doc.ic.ac.uk/∼ajd/Scene/Release/vw34.tar.gz

Table 1: Time budgets and their relationships in the experimental setup (MGT stands for Maximum Generation Time and MET stands for Maximum Evaluation Time).

Budget (ms)	Hz	MGT (budget×0.667)	MET (budget×0.333)	MNHG (budget×5)
16.667	60	11.116889	5.550111	83.335
33.333	30	22.233111	11.099889	166.665
40	25	26.68	13.32	200
50	20	33.35	16.65	250
66.667	15	44.466889	22.200111	333.335
100	10	66.7	33.3	500

4 EXPERIMENTS

The experiments were performed according the simulation methodology described in the previous section. The hardware used in the experiments consisted of one powerful computer and one low-budget computer for the current standards. The first computer is equipped with an Intel Core i7 second generation processor running at 3.07 GHz that, despite having 8 cores, had the processes launched in a Windows 7, single-threaded console to perform all the processing. The second computer is equipped with a AMD A6-3400M processor quad core CPU, running at 1.40 GHz, in which we also launched the processes in a Windows 7, single-threaded console.

We made the option for using single-thread process in order to make simple and fair comparisons. Since only one thread is available, it is simpler to reproduce the simulations under the same conditions and check the performance even in distinct hardware setups and/or architectures. This also minimizes the possible impact of any other process that may be started by the operating system.

As mentioned in the previous Section, four experiments were performed. The first experiment was used only to validate the results our approach. The results of the other three experiments are now presented in a total of three tables. These tables include information about time budget, GPR block size, GPR rejection fraction, the average number of generated hypotheses, and the average error for both GPR and P-RANSAC, and for both forward and side-wise motions.

The Table 2 shows the results of the experiments for both forward and side-wise motions with the original MNHG value run on the more powerful device, while Table 3 shows the results of the same experiments but with the modified MNHG running in the same device. Finally, Table 4 shows the results with the modified MNHG, but when run in the low-budget device.

5 DISCUSSION

The results show that our hypothesis is correct, i.e., that one can find parameters that may balance the preemption scheme and produce better results than by using the standard values originally selected for P-RANSAC. They also demonstrate that our assumption about MNHG is plausible, since the TError presented small variability in some cases and a bit wider in others.

The results of the experiments show that when the GPR parameters are well tuned, the variability of its results is smaller than the P-RANSAC results. We can also infer from the results that the GPR accuracy was consistent in every scenario. That does not means that GPR produced the best results in all tests, but that its flexibility allows it to be tuned so it can be used for a particular configuration of application, time constraints and hardware.

We plotted the average errors for the forward and the side-wise motions for both algorithms. In Figures 1 and 2 we show the results for the more powerful machine. We did not include the results of P-RANSAC with the modified MNHG scenario in order to preserve the scale of the graph, since it reached values around five times larger than the other ones.

Figures 3 and 4 compare the results of GPR and P-RANSAC in the modified MNHG scenario in the low-budget device. Actually, the results of P-RANSAC in the modified MNHG scenario are similar for both devices, thus, the average translational errors for the more powerful machine that were not included in Figures 1 and 2 can be inferred from the results shown on Figures 3 and 4.

Figure 1 confirms the possibility of achieving similar results by using GPR with different parameters selection than of the standard setup, and that the GPR with modified value for MNHG showed less variation in the average error relative to the variation of time budget.

The first charted point of Figures 1 and 2 present

Table 2: Experiments with original value of MNHG. (FtR stands for Fraction to Remove).

			Forward Motion			side-wise Motion		
Budget (ms)	GPR Block	FtR (%)	Average Generated Hypotheses	Average GPR TError (°)	Average P-RANSAC TError (°)	Average Generated Hypotheses	Average GPR TError (°)	Average P-RANSAC TError (°)
16.667	20	20	478.906	14.8458	8.31558	360.962	5.11734	5.66775
33.333	30	20	501.447	8.65899	5.94851	500.561	5.07457	4.8266
40	40	30	501.348	6.48244	6.3134	500.978	4.73236	5.08546
50	60	40	501.389	6.3561	6.15591	501.006	5.22747	4.96326
66.667	60	40	501.419	6.25041	5.92838	500.978	5.31673	4.64777
100	60	40	501.462	6.28521	16.925	501.039	5.38165	10.7852

Table 3: Experiments with modified MNHG. (FtR stands for Fraction to Remove).

			Forward Motion			side-wise Motion		
Budget (ms)	GPR Block	FtR (%)	Average Generated Hypotheses	Average GPR TError (°)	Average P-RANSAC TError (°)	Average Generated Hypotheses	Average GPR TError (°)	Average P-RANSAC TError (°)
16.667	45	45	85.463	10.3403	46.6469	84.989	6.98301	55.4828
33.333	50	65	168.352	7.70724	45.871	167.978	5.95346	53.8409
40	50	65	201.395	7.36652	44.975	200.974	5.6052	53.8277
50	45	50	251.401	7.35803	42.8753	250.966	5.33252	52.456
66.667	45	50	335.423	6.67942	44.4866	334.997	4.91618	56.4664
100	45	50	501.37	6.37809	45.0267	500.984	4.79571	55.9154

Table 4: Experiments with modification in the maximum number of hypotheses to generate, forward motion. (FtR stands for Fraction to Remove).

			Forward Motion			side-wise Motion		
Budget (ms)	GPR Block	FtR (%)	Average Generated Hypotheses	Average GPR TError (°)	Average P-RANSAC TError (°)	Average Generated Hypotheses	Average GPR TError (°)	Average P-RANSAC TError (°)
16.667	20	20	85.36	15.1992	45.763	85.029	7.51641	56.8997
33.333	30	20	168.505	11.4411	46.0835	167.963	6.11811	52.2122
40	40	40	201.454	8.08279	44.578	200.989	5.62041	56.5226
50	60	40	251.445	7.59762	44.9756	250.954	5.80402	51.546
66.667	60	40	335.426	6.79015	44.9835	334.985	5.61065	51.5363
100	60	40	501.355	6.26112	43.1999	500.959	5.29221	53.072

an isolated degenerated behavior of P-RANSAC. In a deeper investigation, we detected that the degeneracy is the effect of the unbalanced amount of hypotheses and the maximum number of tests that leaded to a scenario in which the algorithm discarded an insufficient amount of hypotheses to enforce the estimation quality.

Figure 2 shows a similar behavior of all three algorithms, except for the GPR with standard MNHG when the time budget is lowered for forward motion. We also found out that the GPR with standard MNHG was more robust for lower budgets in side-wise motion, since it did not show any significant error increase.

We have shown that one can use GPR with different hardware configurations and constraints, in cases where the P-RANSAC fails to produce good results. An important detail is that the specific balance we achieve here is related to the hardware configuration used. Different hardware configurations may, and most certainly will, have a different set of optimal parameters.

It is necessary to mention the two main obstacles for tuning the GPR parameters. As would be expected, the first and bigger difficulty is the combinatorial nature of optimization. We used a semi-automated brute force scheme to find promising combination of parameters yielding good results.

The other problem is that a balanced setup for a particular hardware configuration is not guaranteed to work properly for a different configuration, leading to the search of a new set of parameters for that particu-

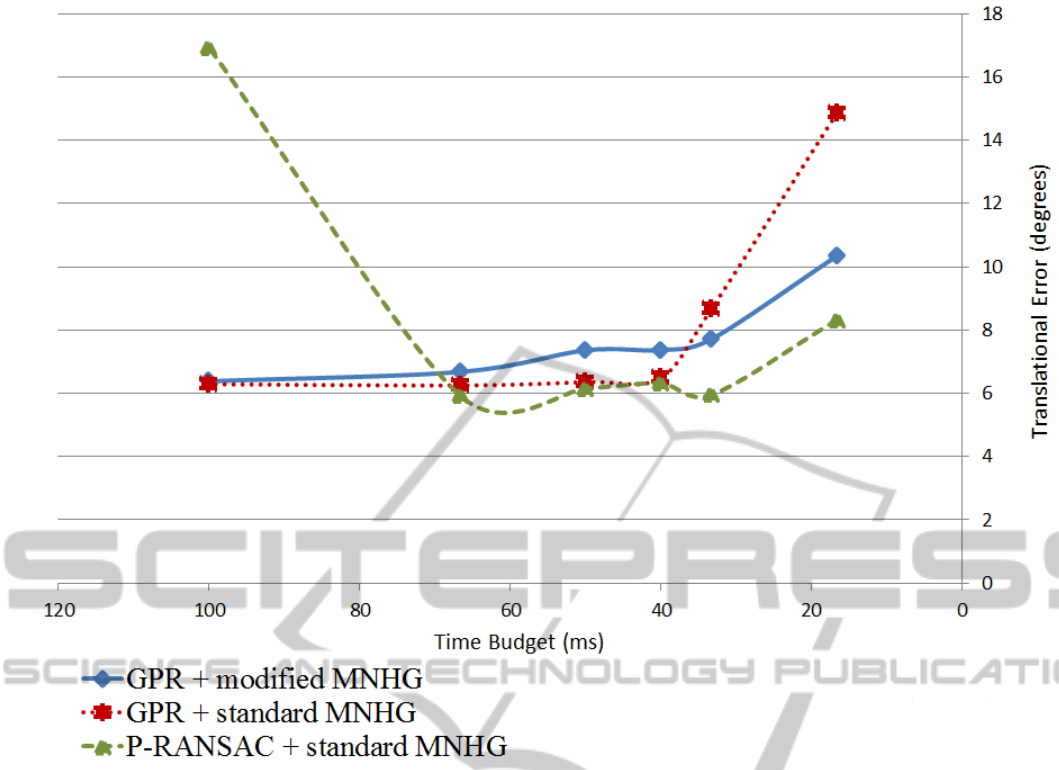

Figure 1: Time budget x Translational error in degrees (forward motion).

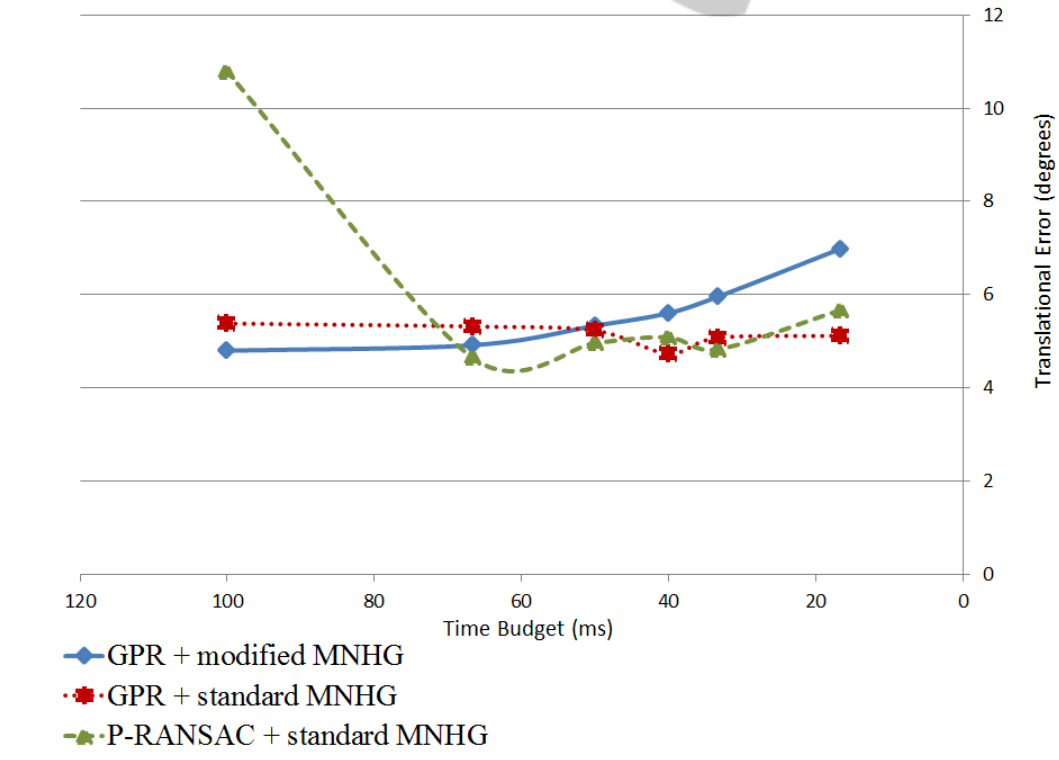

Figure 2: Time budget x Translational error in degrees (side-wise motion).

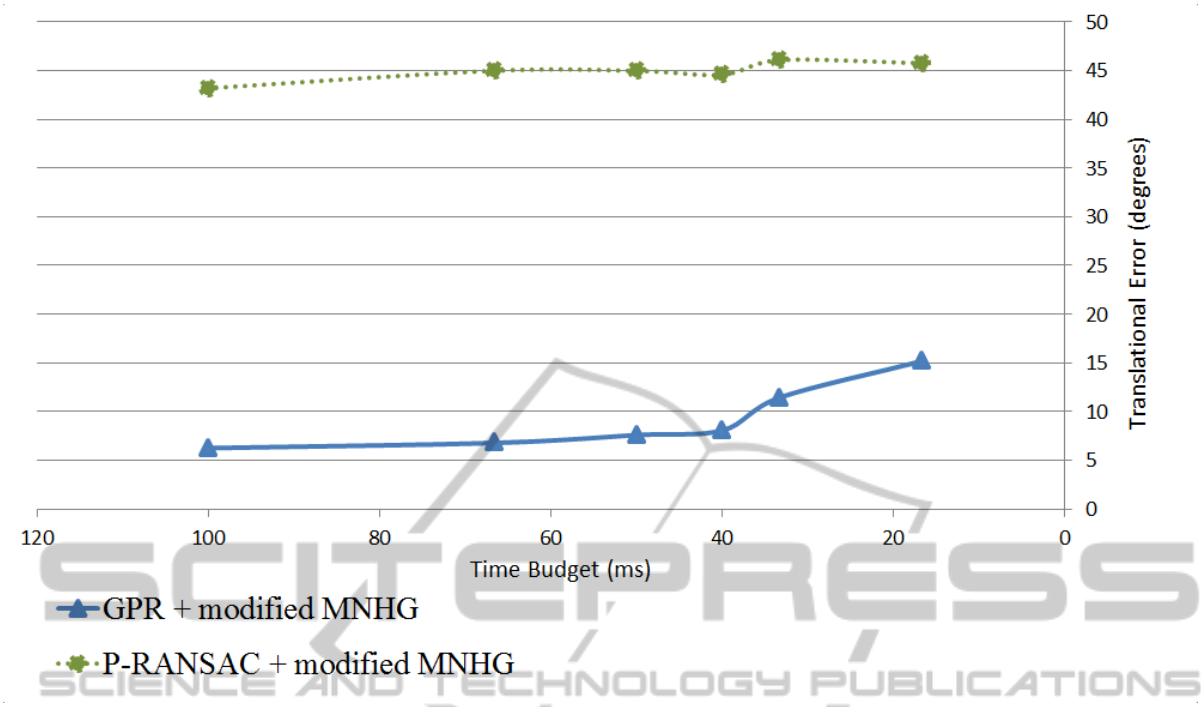

Figure 3: Time budget x Translational error in degrees (forward motion) in a modest device.

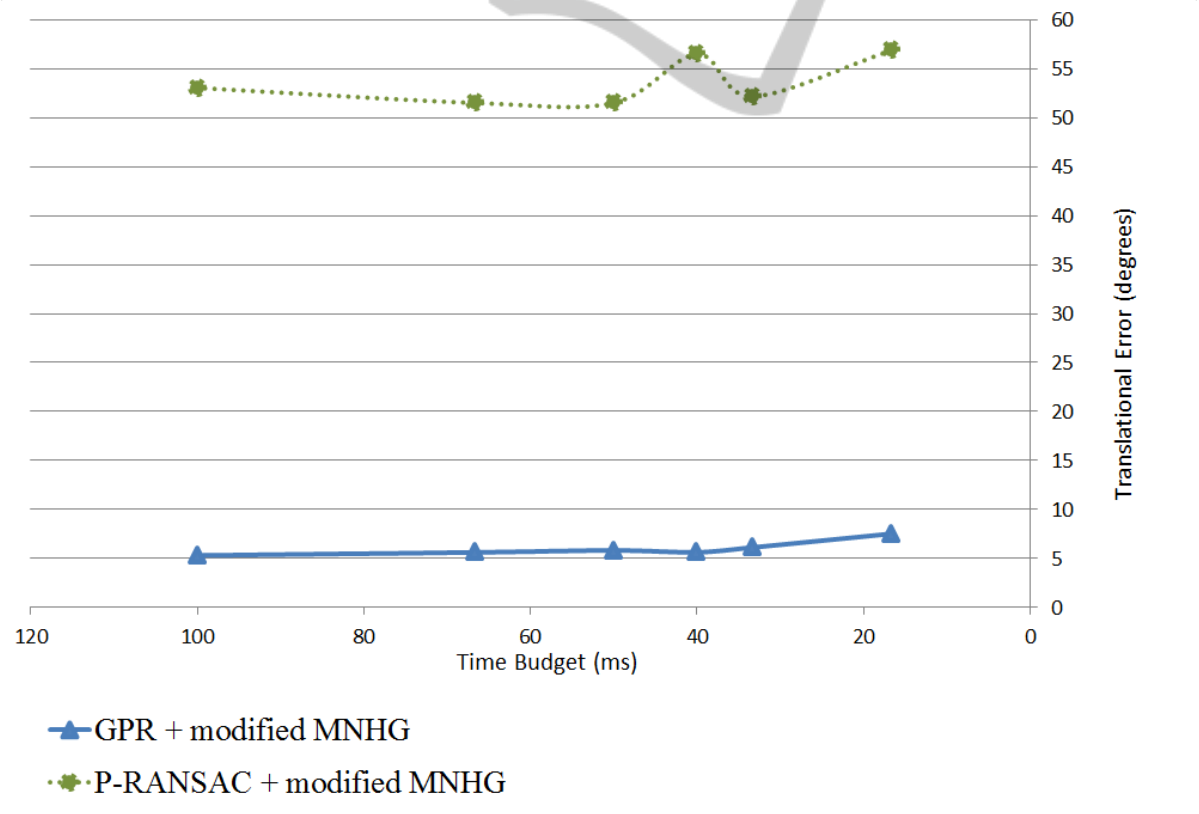

Figure 4: Time budget x Translational error in degrees (side-wise motion) in a modest device.

lar hardware configuration. However, once the budget and the parameters have been defined for a particular hardware configuration, they can be used for any instance of such system.

6 CONCLUSIONS

In this paper, we proposed the usage of the Generalized Preemptive RANSAC (GPR), that can be seen as a generalization of the Preemptive RANSAC (P-RANSAC), and can be applied successfully to different hardware configurations, specifically to low-budget hardware configurations.

We concentrated in validate our approach in a controlled scenario in order to assure a high level of confidence in the results since we have the ground truth as a reference for the comparisons along the several experiments we have performed. The use of two-view geometry and a single model (cloud of points projected in two synthetic frames) demonstrated to be adequate to our needs.

We tested the algorithms on synthetic data, analyzing the average translational error produced for several time budgets, in order to simulate different time constraints required by applications with different objectives. The flexibility of the GPR allowed it to maintain lower average error rates even in the low-budget hardware configuration used. This means that the flexibility of GPR allows a number of computer/robot vision applications to be developed even when using modest hardware setups.

Future works include a deeper study of the parameter setup to correlate the information and try to estimate a model that can give a range of values for the model's parameters in order to decrease the need for human intervention and decrease the amount of time needed for the parameter setup. At the end, we expect to deliver a system able to run a calibration step in the hardware setup that aids the determination of adequate parameters for a given time budget in the selected device. New experiments are been conducted using real pairs of images to compare the performance of both approaches under such conditions.

We also plan to investigate the finding of several motion models in a single image pair, as well as combining some hypotheses generation methods (e.g. 5-point, 8-point and homography) in the generation step in order to promote diversity of hypotheses. This will make GPR adapted to work with a variable number of models. We expect that this will provide more reliability in the estimation since each model's motion may be best estimated by distinct types of hypotheses, i.e. computed by distinct generation methods, such as backgrounds mapped by homographies and other objects mapped by the 8-point algorithm in forward motion (which surpasses 5-point in such conditions (Šegvić et al., 2007a)).

Finally we plan to perform tests with other hardware platforms, specially those low budget processors which are been used in autonomous robot vision implementations.

ACKNOWLEDGEMENTS

The authors would like to express gratitude to Professor Siniša Šegvić from the University of Zagreb for all his help and for providing us with his simulation code, on which we built on. Bruno M. Carvalho is supported by FAPERN/CNPq PRONEM Grant.

REFERENCES

Chum, O. and Matas, J. (2005). Matching with PROSAC - Progressive Sample Consensus. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, pages 220–226, Washington, DC, USA. IEEE Computer Society.

Chum, O. and Matas, J. (2008). Optimal Randomized RANSAC. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(8):1472–1482.

Fischler, M. A. and Bolles, R. C. (1981). Random Sample Consensus: A Paradigm For Model Fitting With Applications to Image Analysis and Automated Cartography. *Commun. ACM*, 24(6):381–395.

Gomes-Neto, S. and Carvalho, B. M. d. (2010). BRUMA: Generalizing All Preemption Functions. In *Proc. IWSSIP*, pages 380–383.

Konouchine, A., Gaganov, V., and Veznevets, V. (2005). AMLESAC: A New Maximum Likelihood Robust Estimator. In *In Proc. Graphicon05*, pages 93–100.

Li, H. and Hartley, R. (2006). Five-Point Motion Estimation Made Easy. In *Proceedings of the 18th International Conference on Pattern Recognition - Volume 01*, ICPR '06, pages 630–633, Washington, DC, USA. IEEE Computer Society.

Michaelsen, E., von Hansen, W., Meidow, J., Kirchhof, M., and Stilla, U. (2006). Estimating The Essential Matrix: GOODSAC versus RANSAC. In *Symposium of ISPRS Commission III: Photogrammetric Computer Vision*, pages 161–166.

Nistér, D. (2003). Preemptive RANSAC for Live Structure and Motion Estimation. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 199–, Washington, DC, USA. IEEE Computer Society.

Nistér, D. (2004). An Efficient Solution to the Five-Point Relative Pose Problem. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(6):756–777.

Raguram, R., Frahm, J.-M., and Pollefeys, M. (2008). A Comparative Analysis of RANSAC Techniques Leading to Adaptive Real-Time Random Sample Consensus. In Forsyth, D., Torr, P., and Zisserman, A., editors, *Computer Vision ECCV 2008*, volume 5303 of *Lecture Notes in Computer Science*, pages 500–513. Springer Berlin Heidelberg.

Rodehorst, V. and Hellwich, O. (2006). Genetic Algorithm SAmple Consensus (gasac) - A Parallel Strategy for Robust Parameter Estimation. In *Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop*, CVPRW '06, pages 103–, Washington, DC, USA. IEEE Computer Society.

Torr, P. H. S. and Zisserman, A. (2000). MLESAC: A New Robust Estimator with Application to Estimating Image Geometry. *Computer Vision and Image Understanding*, 78:2000.

Vedaldi, A., Jin, H., Favaro, P., and Soatto, S. (2005). KALMANSAC: Robust Filtering by Consensus. In *ICCV*, pages 633–640. IEEE Computer Society.

Šegvić, S., Schweighofer, G., and Pinz, A. (2007a). Influence of numerical conditioning on the accuracy of relative orientation. In *CVPR*. IEEE Computer Society.

Šegvić, S., Schweighofer, G., and Pinz, A. (2007b). Performance evaluation of the five-point relative pose with emphasis on planar scenes. In *Performance Evaluation for Computer Vision*, pages 33–40, Austria. Workshop of the Austrian Association for Pattern Recognition.