FPGA Based Massively Parallel Hybrid Architecture for Parallelizing RRTs

Gurshaant Singh Malik
International Institute of Information Technology, Hyderabad, India
Email: garymalik8080@gmail.com

Krishna Gupta
International Institute of Information Technology, Hyderabad, India
Email: Krishna.gupta@research.iiit.ac.in

Raunak Dharani
North Carolina State University, USA
Email: rpdharan@ncsu.edu

K. Madhava Krishna
International Institute of Information Technology, Hyderabad, India
Email : mkrishna@iiit.ac.in

***Abstract*— Field Programmable Gate Arrays(FPGA) exceed the computing power of software based implementations by breaking the paradigm of sequential execution and accomplishing more per clock cycle by enabling hardware level parallelization at an architectural level. As has been proved in already published research works, introducing parallel architectures for a computationally intensive algorithm like Rapidly Exploring Random Trees(RRT) will result in an exploration that is fast, dense and uniform. FPGA based combinatorial architecture, which is one of the already published research works, delivers superlative speed-up but consumes very high power. The second of the already published research works, FPGA based hierarchical architecture, delivers relatively lower speed-up with acceptable power consumption levels. To combine the qualities of both, a hybrid architecture, that encompasses both combinatorial and hierarchical architecture, is designed. To determine the design parameters of the hybrid architecture, a cost function comprised of fundamentally inversely related speed-up and power parameters, as mentioned above, is formulated. This maximization of cost function, with its associated constraints, is then mathematically solved using a modified branch and bound, that leads to optimal deduction of the design parameters of the hybrid architecture. Via empirical experiments, it is observed that this hybrid architecture delivers the highest performance-per-watt out of the three architectures for differential, quad-copter and fixed wing kinematics, in environments of varying geometric complexity. The empirical experiments also confirmed that the hybrid architecture is scalable with 1.) Increase in the environment's geometric complexity and 2.) Increase in kinematic complexity of the robot.**

***Index Terms*—FPGA, Mobile Robotics, RRT, Parallelization, Hardware, Exploration.**

I. Introduction

A. Previous Works

During the last decade and a half, as computer power has increased, sampling-based path planning algorithms, such as rapidly exploring random trees (RRT), have been shown to work well in practice and possess theoretical guarantees such as probabilistic completeness. A significant amount of research effort has gone into improving the performance of RRTs. From an architectural standpoint, recent research efforts have been directed towards parallelizing RRT [1-4]. Out of these, distributed RRT [1] proffers the use of MPI for inter-module communication between multiple RRT modules to maintain data sanity, at the cost of inter-RRT scheduling. K-distributed [2] reduces this scheduling by lowering the amount of inter-RRT communication, at the cost of a less uniform exploration. However, FPGA based combinatorial [3] and hierarchical [4] architectures, have already been shown to perform better than these implementations. The authors highly recommend the readers to familiarise themselves with the hierarchical and combinatorial architecture to better appreciate the current work.

B. Advantage of FPGA in the Field of Robotics

FPGA enables delivery of tightly packed, energy efficient infrastructures adept in fast real time performance [5-8]. Unlike a software effort in parallelization [1], [2], an FPGA allows gate level control of system architecture for parallelizing RRT. This allows the designer to tap the potential of hardware design, allowing control over minute details of arithmetic design, real time parallelization, and pipe-lining of sequential

Manuscript received July 1, 2017; revised August 21, 2017

doi: 10.18178/ijmerr.6.6.467-475

processes. The hardware level flexibility afforded by an FPGA results in parallel RRT architectures that are not only fast, but also small and power efficient.

C. Overview and Necessity of Hybrid Architecture

Parallel RRTs refers to multiple RRT modules (RRT algorithm functions) working in parallel to explore the environment. FPGA based combinatorial and hierarchical architectures have already been shown to perform better than state of the art parallel RRT architectures like distributed [1] and K-distributed [2]. While the speed-up of combinatorial is very high, the biggest handicap of the combinatorial architecture is its very high power consumption. This renders the combinatorial architecture's deployment across mobile robots, which typically are tightly constrained with respect to battery capacity, very limited and unfeasible. The hierarchical architecture consumes little to moderate power but has relatively lower speed-up compared to combinatorial. This makes the deployment of hierarchical architecture in environments that demand a very quick reaction, particularly tricky and impractical. Hence, an architecture with maximum speed-up and minimum power consumption is highly desired. To converge towards this theoretical ideality, a flexible and malleable hybrid architecture is designed. In a N parallel RRT hybrid architecture, M RRT modules are allotted to combinatorial and $N-M$ RRT modules are allotted to hierarchical. This means that M RRT modules work in parallel via combinatorial architecture and the remaining $N-M$ RRT modules work in parallel via hierarchical architecture. The determination of M is mathematically calculated, with the calculations centered around maximization of a cost function, using a set of constraints explained in later sections. The subsequently designed hybrid architecture is then tested successfully for scalability across robotic kinematic complexity (differential, quad-copter and fixed wing kinematics) and geometric environment complexity.

II. Challenges in Parallelizing RRT

Since RRT involves randomized exploration of the environment, ours and many proposed algorithms [1], [2] use the principle of exploratory decomposition [9] as their foundation. Fig. 1 provides an overview of this design philosophy.

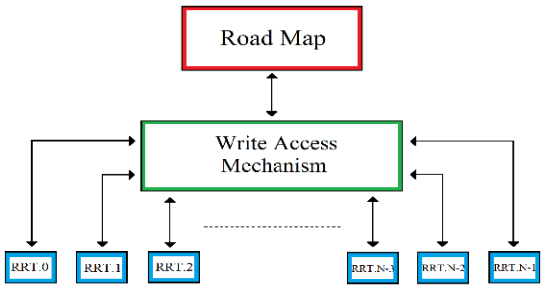


Figure 1. Schematic illustration of exploratory decomposition.

In such a parallel RRT design, an important issue is to decide the write access mechanism that integrates the data from multiple RRTs and then updates the global explored map. There are 2 general philosophies: 1.) Distributed and 2.) Shared. The distributed philosophy employs a scheme by which each RRT will have its own local explored map. As a result, changes made by it to its local explored map will have no effect on other RRT's local explored maps. Hence, as shown in Fig. 2, we need a 'mediator' system that updates each RRT's local explored map to changes made by other RRTs. This will incur significant inter-RRT communication time in case of large scale parallelization.

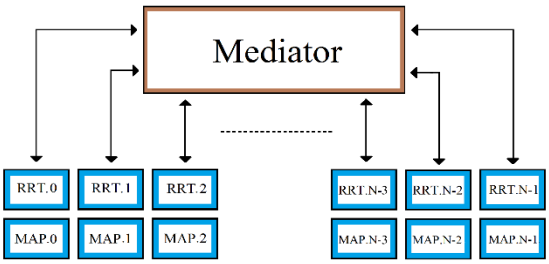


Figure 2. Schematic illustration of distributed philosophy

The shared design philosophy, shown in Fig. 3, allows all the RRTs to have access to the same global explored map. Hence, there is virtually no inter-RRT communication. But, since all RRTs will have access to the same global explored map, large scale parallelization, without scheduling, can geometrically increase traffic on global address space, leading to data collisions.

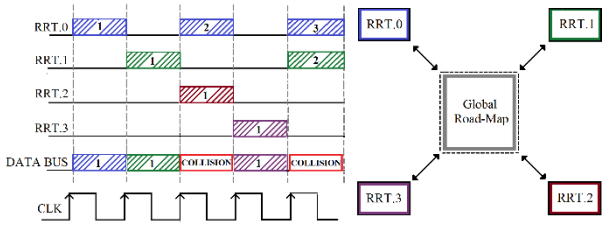


Figure 3. Schematic illustration of the Shared design philosophy

Conventional software implementation of parallel RRTs [1], [2] solve the problem of this contentious relationship between scheduling and data integrity by using frameworks such as MPI, STAPLE etc. However hardware implementations, owing to RTL level optimizations, have been shown to perform significantly better than their software counterparts in the case of parallel RRTs [3], [4]. As shown in Fig. 4, FPGA based hierarchical architecture has a respectable speed and power cost function.

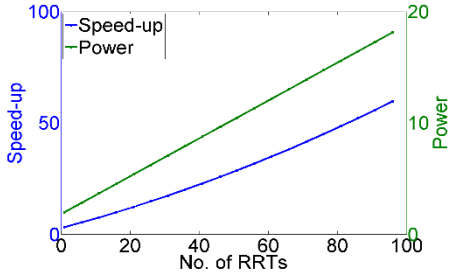


Figure 4. Hierarchical: Speed-up and Power plots

As shown in Fig. 5, combinatorial architecture has a very high speed-up but also a very high power cost function.

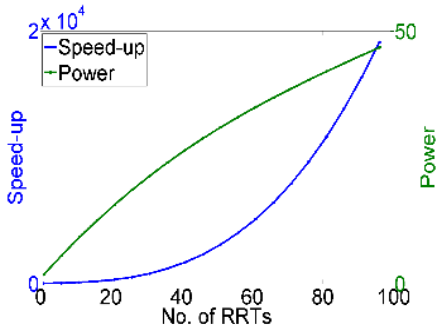


Figure 5. Combinatorial: Speed-up and Power plots

Ideally, a parallel RRT architecture should have a speed-up capability similar to combinatorial and power consumption levels similar to hierarchical. The next section delineates on this requirement with the proposed hybrid architecture.

III. Proposed Hybrid Architecture

To enable intelligible understanding of the proposal, this section is further divided into 3 subsections. 1.) Hybrid Architecture's hypothesis, 2.) Hybrid Architecture's design and mathematical variables, 3.) Cost Function, to calculate the hybrid architecture's variables, strictly constrained by a set of intelligent, FPGA platform sensitive conditions.

A. Hypothesis

Owing to the probability reliant exploration of RRT, accurate prognosis of data arrival time is an ambiguous task. Hence N RRTs working in parallel can result in 2^N possible cases during a write window to the global road-map. In order to theoretically rationalize the hypothesis behind the hybrid architecture, it is important to characterize the architecture, speed-up and power consumption levels of the hierarchical and combinatorial architectures, in chronological order.

In hierarchical, as shown in Fig. 6, the data stems from the RRT modules and flows through higher levels of hierarchy to reach the global map. P stands for POLL, F stands for FIFO. At the deepest level, P0 chronologically polls RRT0, RRT1. P1 polls RRT2, RRT3 and so on. Going up, F00 polls P0, P1. F01 polls P2, P3 and so on. Going up a level, F10 polls F00, F01 and F11 polls F02, F03. Finally, at the highest level, the global road-map is updated by F10 and F11. At all levels, chronological polling for data by parent module preserves data integrity but the architecture is still weighed down by the scheduling that prevails amongst child modules of the parent modules.

Eqs. 1 and 2 present the speed-up and power consumption levels respectively for the hierarchical architecture, extracted out of the data for speed-up and power available with the authors, for N parallel RRT modules. Curve fit method is used to formulate the equations.

$$S(N) = 0.0019N^2 + 0.41N + 2.8 \quad (1)$$

$$P(N) = 0.17N + 1.8 \quad (2)$$

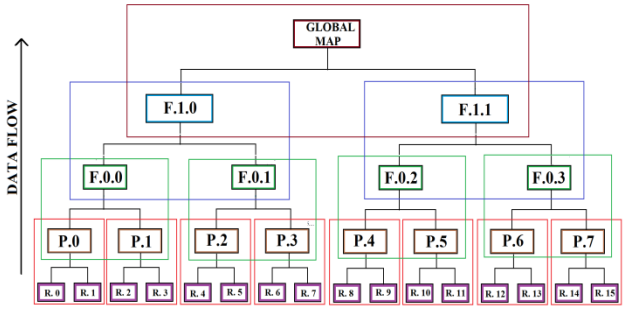


Figure 6. Hierarchical architecture

In combinatorial, as shown in Fig. 7, the first part of the architecture is a multi-port random access memory with the ability to handle [0, N] variable, asynchronous write and/or read transactions during a write and/or read window. The second part of the architecture is a combinatorial circuit that ascertains the current case of the 2^N cases during the write window and feeds the appropriate write control signals to the memory. This allows each of the N RRTs to have access to the write window with zero latency/scheduling since this write access mechanism combinatorially accounts for all the possible 2^N cases. Eqs. 3 and 4 present the speed-up and power consumption levels respectively for the combinatorial architecture, extracted out of the data for speed-up and power available with the authors, for N parallel RRT modules. Curve fit method is used to formulate the equations.

$$S(N) = 0.021N^3 + 5.7N + 3.3 \quad (3)$$

$$\mathrm{P}(N) = 1.6 * 10^{-5} * N^3 - 0.0048N^2 + 0.79N + 1 \quad (4)$$

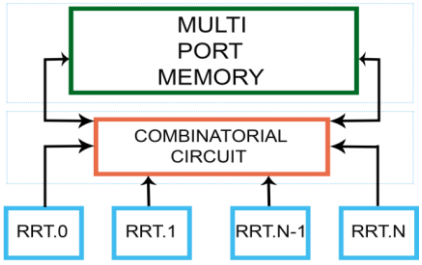


Figure 7. Combinatorial architecture

Comparing Eqs. 1 and 3, plotted in Fig. 8(left), architecturally, combinatorial aggressively out-throttles hierarchical. But on the other hand, comparing Eqs. 2 and 4, also plotted in Fig. 8(right), hierarchical is of a much more clement nature in power consumption. It should be noted that speed-up directly controls the accelerated capability of the system. That is, how fast a map is explored. Mobile robots typically are constrained by a small battery. Hence, quantitatively, a maximal bound that is very small in magnitude needs to be placed on power consumption levels. Theoretically, it can be

concluded with confidence that an architecture that behaviorally is analogous to hierarchical in terms of power consumption and analogous to combinatorial in terms of speed-up is ideal. Hence, a hybrid architecture that combines these properties of hierarchical and combinatorial is hypothesized.

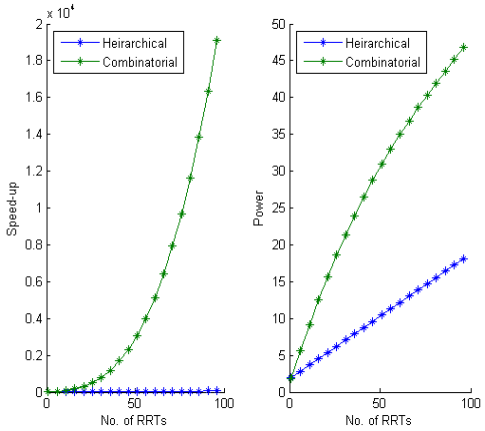


Figure 8. Speed-Up and Power: Combinatorial VS Hierarchical

B. Hybrid Architecture

Fig. 9 provides a top level architectural view of the hybrid architecture for N parallel RRT modules. Consequent delineation follows. The critical design philosophy instructs the division of these N RRT modules into 2 parts: 1.) M RRT modules are aligned to follow the combinatorial architecture and 2.) Remaining N-M RRT modules are aligned to hierarchical architecture. Hence, by varying M, the architecture can be made to cover the entire behavioral spectrum, with the extreme being hierarchical for M=0 and combinatorial for M=N. Eqs. 5 and 6 mathematically quantify this property.

$$S_{Hybrid}(M) \rightarrow [S_{Hier}(M=0), S_{Combi}(M=N)] \quad (5)$$

$$P_{Hybrid}(M) \rightarrow [P_{Hier}(M=0), P_{Combi}(M=N)] \quad (6)$$

$$Total_{Hier}(M) = N - M \quad (7)$$

$$Total_{Combi}(M) = M + 1 \quad (8)$$

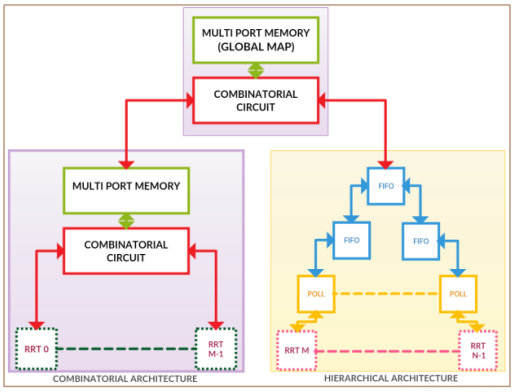


Figure 9. Schematic illustration of hybrid architecture

As shown in Fig. 9, M RRT modules explore the map in parallel via combinatorial architecture and the remaining N-M RRT modules explore the map in parallel via hierarchical architecture. These 2 exploration results are then combined together to form the global explored road-map via combinatorial block (Hence, total M+1 blocks). Eqs. 7 and 8 describe the number of combinatorial and hierarchical blocks respectively. For a given value of N, the entire behavior of this architecture is administered by the variable M. The next subsection outlines the mathematical formulas and constraints critical to the deduction of the variable M.

C. Cost Function

Owing to the mathematical complexity and volume involved, for unabridged understanding of the concept, this section is further divided into 4 subsections : 1.) The cost function, 2.) Set of constraints to generate a singular, optimized solution, 3.) The computation strategy to generate the solution and 4.) Arbitration of the tagging of M RRT modules out of the total N RRT modules.

1) Cost function

Eqs. 1, 2, 3 and 4, clearly manifest the irreconcilable nature of maximality and minimality between speed-up and power consumption since both of them are proportional to the number of parallel RRT modules. That is, speed-up cannot be maximized in conjunction with minimized power consumption. Hence, instead of solitary maximization of speed-up and minimization of power consumption, we aim to maximize the cost function given in Eq. 9, with individual terms delineated in Eqs. 10 and 11. While adjudicating about the formulation of the cost function, it was observed that the form S/P was biased towards minimizing power whereas S+1/P was moderate in nature. As described in Eq. 8, it should be noted that for M combinatorial RRT modules, there exist M+1 combinatorial blocks.

$$J_{Hyb}(M) = S_{Total}(M) + \frac{1}{P_{Total}(M)} \quad (9)$$

$$S_{Total}(M) = S_{Hier}(N-M) + S_{Combi}(M) \quad (10)$$

$$P_{Total}(M) = P_{Hier}(N-M) + P_{Combi}(M+1) \quad (11)$$

2) Set of constraints

The cost function is maximized subject to the following constraints:

- Naturally, M must be a positive integer

$$M > 0, \in I \quad (12)$$

- M must not exceed N

$$M \leq N \quad (13)$$

- Sensitive to robotic platform's battery endurance capability, the designer decides

how much maximum power (ω) the architecture can consume.

$$P_{Hyb}(M) \leq \omega \quad (14)$$

3) Mathematical solver

To solve for M, we aim to maximize the cost function, as previously described in Eq. 9. For this, Branch and Bound [10], a systematic solver for optimized integer solutions, is used. Branch and bound has the ability to accept non-linear optimization problems as inputs, as is the case with the formulae. The generic algorithm of branch and bound follows.

Algorithm 1: Branch and Bound Algorithm

```
E  : nodepointer
H  : heap
E := new(node) ;
while ~true do
    if E is final leaf then
        Return.Solution ;
    Expand(E) ;
    if H is empty then
        Return.NoSolution ;
    Delete.Top(H) ;
```

4) Tagging of M RRT modules

Post the calculation of M, the next step is to decide that out of the N RRT modules, which of the M RRT modules to ascribe to combinatorial architecture and the remaining N-M to hierarchical architecture. It should be noted that the calculation of M has already been done in the previous step. This step is to determine the tagging of the RRT modules to combinatorial or hierarchical. This identification, as described in Eq.15, is driven by the user's decision about the approximate average map area (α) each of the combinatorial M RRT modules should explore.

$$\sum_{i=1}^{M} \frac{A_{Combi}(i)}{M} \geq \alpha \quad (15)$$

$$A_{Combi}(i) = \frac{1}{\sum_{r=1}^{grids} d_r} * A_{Map} \quad (16)$$

$$d_r = BFS.distance(grid_r, start.node_i) \quad (17)$$

As shown in Fig. 10, the map is divided into high resolution grids and the area is calculated, as described in Eqs. 16 and 17, by summing the distance between that RRT module's starting node and each grid. It should be noted that the user has the flexibility of intelligently [11] or randomly choosing the starting nodes. For the current experimental setup, a value of α was so chosen that the RRT nodes with the top M areas were allotted to combinatorial architecture.

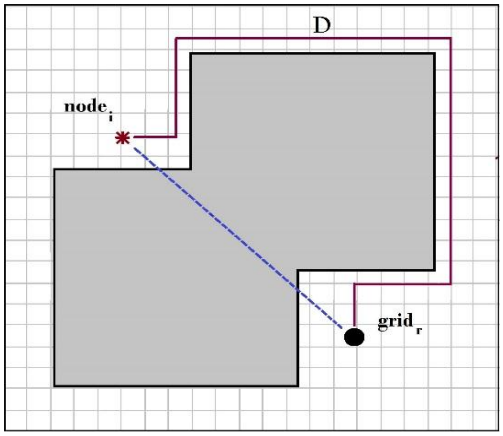


Figure 10. Calculation of distance for area estimation

IV. FPGA Implementation

The design platform, Zedboard, uses the Zynq-7000 SOC, with system parameters given in table below. For our design, the cartesian coordinates are represented as 32bit long, fixed-point, 2's complement binary strings where the 24 MSB represent the integer part and the 8 LSB represent the fractional part. This representation provides an incremental resolution of 0.00390625 in decimal format. The geometrical angle is represented as a 16bit long, fixed point, 2s complement binary string where the 3 MSB represent the integer part and the 13 LSB represent the fractional part, affording an incremental resolution of 0.00012207 radians.

TABLE I. Parameters of Design Platform

System		Parameters	
LUT	17600	**BRAM(Mb)**	2.1
Logic Cells	2800	**DSP48E1**	80
CLB FF	35200	**Area(inch^2)**	6.5*5.9

Implementation breakdown, in a bottom to top manner, of each module follows. For the reader's convenience, the hybrid architecture is shown again in Fig. 11.

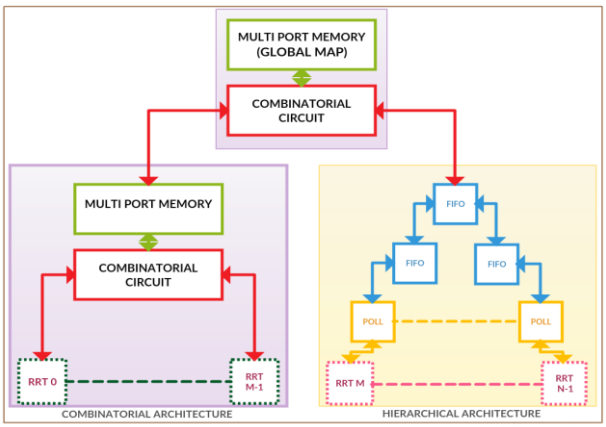


Figure 11. FPGA implementation: Hybrid architecture

A. RRT Module

A pseudo-random number generator generates a random state for the mobile robot in use. We use the box [12] method to find the nearest node. Deployment of DSP48E1 slices minimizes the time complexity of distance computation. CORDIC cores are used for computation of trigonometric functions. DSP slices are then used for kinematic extension.

B. POLL

The POLL is implemented as a sequential Finite State Machine (FSM). Isochronal cyclic polling of child RRT modules germinated by rising edge of clock leads to capture of data bus by one of the children, which then transfers its generated nodes via write-acknowledge mechanism.

C. FIFO

Built in FIFO resources to create high performance, area optimized FIFO module were used. The First-Word-Fall-Through is chosen as the mode of operation for the FIFO interface.

D. Combinatorial Circuit

For N RRTs, the 2^N possible cases and the corresponding control signals of the multi-port memory are mapped to cascaded look up tables (LUTs). An N bit string, where each bit corresponds to a RRT, is used as input. A '1' bit means that the corresponding RRT is requesting access and a '0' bit means otherwise. The outputs of this module are the control signals of the multi-port memory.

E. Multi-Port Memory

With a global address space, the multi-port memory is implemented as a heap of M distributed, single channel memories, each of size (400*F)/M KB, where F is the number of degrees of freedom of the robot and M is the number of RRTs. The read and write channels are designed asynchronous to enable independent read and write transactions. Auxiliary multiplexers on the read and write channels apportion the global address space to local address spaces.

V. Results

As shown in Fig. 12, the experimental setup was planned to quantify the architecture across 3 parameters:

- *Performance-per-watt*
- *Scalability across map's geometric complexity*
- *Scalability across kinematic complexity*

Deployment was done across:

- Differentially Steered Firebird V(Actual Run)
- Fixed Wing Aircraft (Simulation)
- Quad-Copter (Simulation)

The test results quantify 3 parameters across kinematic and geometric complexity: 1.) Speed-Up, 2.) Power consumption and 3.) Performance-per-watt.

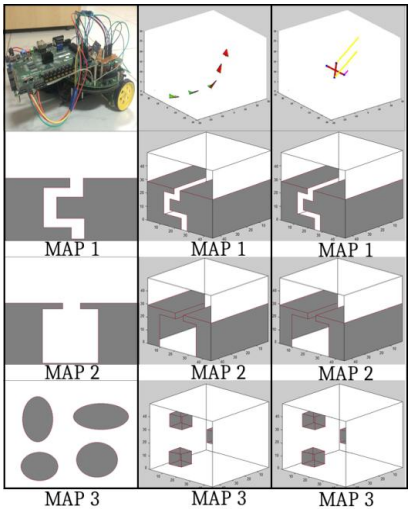


Figure 12. Left Column: Differential, Middle: FWA, Right: Quad-Copter

As described in Eq. 18, speed-up refers to ratio of the time taken by 1 module to complete a particular task, compared to the time taken by N parallel modules, to complete the same task. Relative to the experimental setup, the equivalent task is to add 10,000 explored nodes by N-parallel RRT modules, initially seeded by a modified K-Means [11], to the map. The time is measured by an interrupt driven counter. Efficiency or performance-per-watt, as defined in Eq. 19, is expected to be the maximum for hybrid architecture.

$$S = T(1)/T(N) \tag{18}$$

$$E = S/P \tag{19}$$

It should be noted that, owing to the probabilistic nature of RRT, each iteration was performed 1000 times to get mean values, which are presented in Fig. 13. Row 1 of Fig. 13 benchmarks the architecture for differential drive, row 2 for quad-copter and row 3 for fixed wing aircraft, across a diverse spectrum of geometrically complex maps. It should also be remembered that the line plot is for performance-per-watt while the speed-up and power consumption levels are highlighted for each architecture in the plot in form of (Speed-Up, Power). Please note that Table I details the speed-up and power consumption levels in the following colors: White=Combinatorial, Cyan=Hierarchical and Yellow=Hybrid. The speed-ups are mentioned in the form (Differential (D), Quad-Copter (Q), and Fixed Wing (F)). Quantitative numbers and qualitative reasoning behind the same follows.

A. Speed-Up

As can be concluded from Table II, the combinatorial architecture, unconstrained from any scheduling between RRT modules, delivers the highest speed-ups across the spectrum of kinematic and geometric complexity of (D=424,Q=441,F=440), (D=472,Q=455,F=459) and (D=423,Q=420,F=421) for $N=64$ for Map 1, 2 and 3 respectively. The hierarchical architecture, coerced by scheduling between RRT modules, comes in at a distant

third with the minimum offered speed-ups. The hybrid architecture, designed as an intelligent hybrid of combinatorial and hierarchical so as to achieve speed-ups that are closer to combinatorial, delivers second highest speed-ups of (D=323, Q=315, F=315), (D=342, Q=322, F=321) and (D=317,Q=302,F=302) for N=64 for Map 1, 2 and 3 respectively. Qualitatively, this is enabled by the maximization of the cost function that aims to maximise speed-up and minimise power consumption. While the speed-up offered is definitely lower than combinatorial, it can be confidently concluded that the hybrid architecture delivers on the hypothesis of near combinatorial emulation.

TABLE II. WHITE: COMBINATORIAL, CYAN: HIERARCHICAL, YELLOW:HYBRID

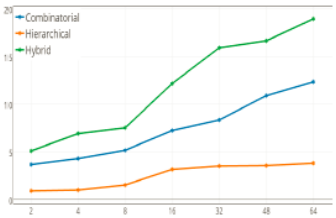
MAPS	N=4	N=16	N=32	N=64
Map 1	27, 20.8, 20.4	97.6, 82.0, 82.3	189.6, 191.3, 190.5	424.1, 441.3, 440.1
Map 2	28.9, 21.8, 22.3	99.4, 85.9, 87.6	213.7, 197.0, 198.2	472.5, 455.6, 459.1
Map 3	21.5, 18.8, 18.5	79.0, 77.5, 77.2	162.3, 182.1, 182.3	423.2, 420.4, 421.3
POWER(W)	6.2, 6.2, 6.3	13.4, 13.5, 13.5	22.7, 22.4, 22.4	34.3, 33.1, 33.3
Map 1	2.2, 3.4, 3.5	9.7, 6.8, 6.9	12.1, 8.5, 8.4	17.0, 14.6, 14.5
Map 2	5.2, 3.7, 3.8	11.9, 8.2, 8.5	13.4, 9.8, 10.1	21, 16.6, 17.3
Map 3	2.3, 3.1, 3.2	8.6, 6.1, 6.0	11.3, 6.8, 6.7	14.2, 11.8, 12.0
POWER(W)	2.1, 2.4, 2.4	3.0, 3.2, 3.3	3.4, 3.2, 3.1	4.4, 4.2, 4.3
Map 1	20.0, 15.6, 15.6	65.2, 58.5, 58.4	142.0, 122.7, 122.3	323.6, 315.7, 315.6
Map 2	21.3, 17.5, 17.8	70.1, 62.0, 62.9	144.6, 132.0, 131.8	342.6, 322.8, 321.8
Map 3	19.5, 12.7, 12.8	61.7, 54.2, 54.0	127.5, 111.0, 110.4	317.4, 302.1, 302.4
POWER(W)	2.8, 3.0, 3.0	5.3, 5.7, 5.8	8.9, 8.4, 8.5	17.0, 17.4, 17.3

B. Power Consumption

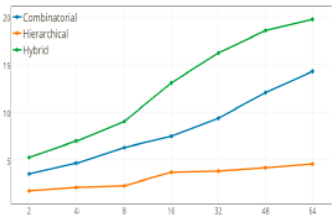
It should be remembered that the static, vector-less power analysis is independent of kinematic and geometric complexity. Analysis of power consumption data, as given in Table I in fourth row of each of the colour segments, reveals that hierarchical, owing to the relatively pliant architecture, consumes the least power levels of (D=4.4W,Q=4.2W,F=4.3W) for N=64. Combinatorial, owing to its expansive combinatorial blocks, is the most power hungry among the three. Hybrid, on the other hand, tries to closely border hierarchical, expending (D=17.0W,Q=17.4W,F=17.3W) for N=64. Qualitative justification behind this behaviour is explicated by the maximization of the cost function that aims to minimise power consumption.

C. Performance-per-Watt

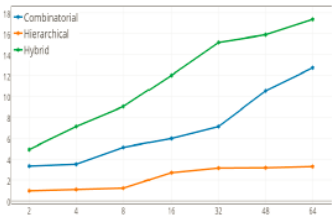
To appreciate the benchmarking primacy hybrid architecture enables over other architectures, it is important to understand that the architecture was designed to maximise speed-up and minimise power consumption concurrently, despite them being antithetical to each other by nature. The maximization of the designed cost function should enable the hybrid architecture to be the most judicious in efficiency or performance-per-watt. This hypothesis is proven true in Fig. 13. As quantized in previous subsections, combinatorial architecture delivers the maximum performance and hierarchical consumes the least amount of power. But, the hybrid tends to closely track the leader in both departments, as already seen. This loose behavioral emulation by the hybrid architecture allows the hybrid architecture to out-throttle both combinatorial and hierarchical in terms of performance-per-watt/efficiency. This out-throttling is observed across the varied spectrum of both geometric as well as kinematic complexities. Hybrid architecture is the most efficient of the 3 architectures with numbers of (D=18.9,Q=18.0,F=18.5), (D=19.7,Q=18.5,F=18.5) and (D=20.4,Q=17,4,F=17.5) for N=64 for Map 1, 2 and 3 respectively. This is true across the complete range of N.



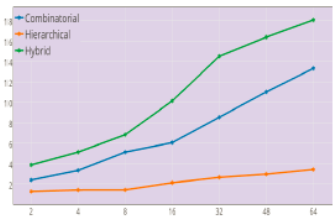
(a) Map 1 : Performance-per-Watt VS No. of RRTs



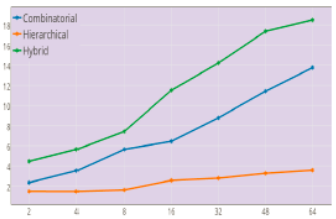
(b) Map 2 : Performance-per-Watt VS No. of RRTs



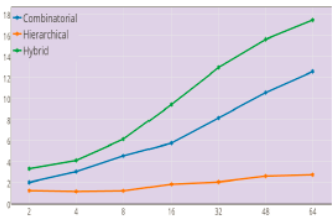
(c) Map 3 : Performance-per-Watt VS No. of RRTs



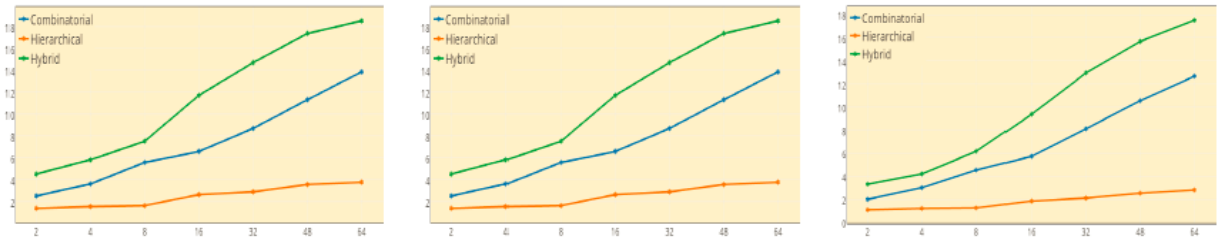
(d) Map 1 : Performance-per-Watt VS No. of RRTs



(e) Map 2 : Performance-per-Watt VS No. of RRTs



(f) Map 3 : Performance-per-Watt VS No. of RRTs



(g) Map 1 : Performance-per-Watty VS No. of RRTs (h) Map 2 : Performance-per-Watt VS No. of RRTs (i) Map 3 : Performance-per-Watt VS No. of RRTs

Figure 13. Comparative analysis of combinatorial, hierarchical and hybrid for the three different test environments across 3 different kinematic test platforms

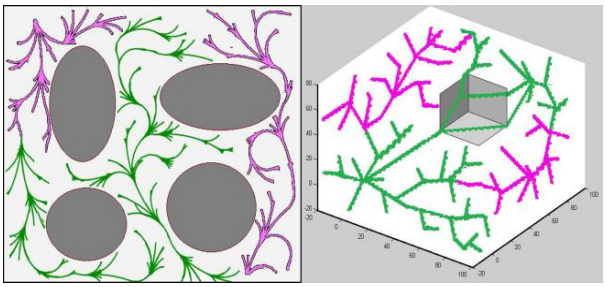


Figure 14. Demo of hybrid architecture for differential and quad-copter

For perceptible understanding, sample runs of differential drive and quad-copter via hybrid architecture is presented in Fig. 14. Pink corresponds to exploration by hierarchical RRT modules and green by combinatorial RRT modules respectively. This benchmarking exercise, performed across varied kinematics and maps, quantitatively proves the 3 qualities of hybrid architecture: 1.) Maximum efficiency/performance-per-watt, 2.) Scalable across map's geometric complexity and 3.) Kinematic complexity.

VI. Conclusion

This paper proffered the hybrid architecture that, apart from benefiting from the inherent parallel abilities of the FPGA, is able to deliver the maximum performance-per-watt amongst state of the art hardware architectures. Quantitative benchmarking of this architecture across different kinematic systems, from land based kinematics to complex aerial kinematics, on maps with tight geometric constraints exhibited the architecture's scalability across kinematic and geometric complexity. As part of our future work, the authors would like to study the scalability of this architecture for non-still, dynamically changing maps with moving obstacles. The authors would also like to extend the optimization methods that enables greater combinatorial speed-up and hierarchical power emulation respectively.

References

[1] Devaurs, T. Sim éon, J. Cort és, "Parallelizing RRT on distributed-memory architectures," *in Proc. IEEE ICRA'11*, 2011, pp–2261.

[2] S. A. Jacobs, N. Stradford, C. Rodriguez, S. Thomas, and N. M. Amato, "A scalable distributed RRT for motion planning," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 5088– 5095.

[3] G. S. Malik, K. Gupta, K. M. Krishna, and S. R. Chowdhury, "FPGA based combinatorial architecture for parallelizing RRT," *in Proc. 2015 European Conference on Mobile Robots*, pp. 1–6.

[4] G. S. Malik, K. Gupta, K. M. Krishna, and S. R. Chowdhury, "FPGA based hierarchical architecture for parallelizing RRT," *in Proc. the 2015 Conference on Advances in Robotics (ACM, 2015)*, pp. 12.

[5] J. Fischer, A. Ruppel, F. Weisshardt, and A. Verl, "A rotation invariant feature descriptor O-DAISY and its FPGA implementation," *in Proc. 2011 IEEE / RSJ International Conference on Intelligent Robots and Systems*, pp. 2365–2370.

[6] J. Svab, T. Krajnik, J. Faigl, and L. Preucil, "FPGA based speeded up robust features," *in Proc. 2009. IEEE International Conference on Technologies for Practical Robot Applications, 2009. TePRA*, pp. 35–41

[7] J. Oberg, K. Eguro, R. Bittner, and A. Forin, "Random decision tree body part recognition using FPGAs," *in Proc. 2012 22nd International Conference on Field Programmable Logic and Applications (FPL)*, pp. 330–337

[8] R. Dubey, N. Pradhan, K. M. Krishna, and S. R. Chowdhury, "Field Programmable Gate Array (FPGA) based collision avoidance using acceleration velocity obstacles," *in Proc. 2012 IEEE International Conference on Robotics and Biomimetics*, pp. 2333–2338

[9] V. Kumar, A. Grama, A. Gupta, and G. Karypis, *Introduction to parallel computing: design and analysis of algorithms* (Benjamin / Cummings Publishing Company Redwood City, CA, 1994)

[10] E. L. Lawler and D. E. Wood, *Operations Research*, vol. 14, p. 699, 1966.

[11] K. M. Krishna, "K-Means derived strategized placement of starting points for parallel RRT's", *in Proc. IIIT Hyderabad* Publications*(IIIT, Hyderabad,2016)*, pp. 145.

[12] M. Svenstrup, T. Bak, and H. J. Andersen, "Minimising computational complexity of the rrt algorithm a practical approach," *in Proc. 2011 IEEE International Conference on Robotics and Automation*, pp. 5602–5607.

Gurshaant Singh Malik was born in Nangal Faizgarh, Punjab, India on 25th November, 1993. He is a former MS by research student at the International Institute of Information Technology, Hyderabad, India. He completed his degree in September, 2016. He also completed his Bachelor of Technology from the same institute in August, 2015. His undergraduate and postgraduate degree were both in the field of Electronics and Communication. He currently works as a SYSTEMS DESIGN ENGINEER at rENIAC. He was also a former SOFTWARE ENGINEER at Xilinx and also INTERNED in the same team before joining as a full-time engineer.

Krishna Gupta is a current MS by research student at the International Institute of Information Technology, Hyderabad, India. He also completed his Bachelor of Technology from the same institute in August, 2015. His undergraduate and postgraduate degree are both in the field of Electronics and Communication.

Raunak Dharani a current MS student at the North Carolina State University, USA. He completed his Bachelor of Technology from the BITS Pilani, Goa campus in August, 2015. His undergraduate was in the field of Electronics and Instrumentation.
He was also a former RESEARCH INTERN at Nanyang Technological University, Singapore. He was also a former SYSTEMS DESIGN INTERN at rENIAC.

K. Madhava Krishna completed his PhD from Indian Institute of Technology, Kanpur in 2001.
He subsequently worked as a POST DOCTORATE FELLOW at the Robotics and AI Lab at LAAS-CNRS, Toulouse, France from 2001-2002. Currently, he is the DIRECTOR of the Robotics Research Center at International Institute of Information Technology, Hyderabad. He is also an ASSOCIATE PROFESSOR at the International Institute of Information Technology, Hyderabad.