Edge Computing for Autonomous Vehicles and Robots

PERCEPTION INSIGHT INTELLIGENCE

Shaoshan Liu
shaoshan.liu@perceptin.io

Fun Facts

What is the best selling robotic processor so far?

Fun Facts

What is the best selling robotic processor so far?

01 Dec 2015 | Geneva

English | French | Italian

STMicroelectronics Ships Billionth ARM-based STM32 Microcontroller and 500 Millionth ST33 Secure Microcontroller

Leading microcontrollers are enablers of both Smart and Secure Things

Geneva / 01 Dec 2015

STMicroelectronics (NYSE: STM), a global semiconductor leader serving customers across the spectrum of electronics applications announced that ST has now delivered more than one billion general-purpose STM32 microcontrollers based on ARM® Cortex® cores. In addition, ST has also passed the 500 million milestone for shipments of ST33 secure microcontrollers built around the ARM SecurCore® SC300 processor.

- Bi-directional operation of up to 60* I/O pins
  - Shared across up to 5 GPIO ports named GPIOA to GPIOF, with up to 16 I/O pins per port
    - All with external interrupt and wake-up capabilities

Fun Facts

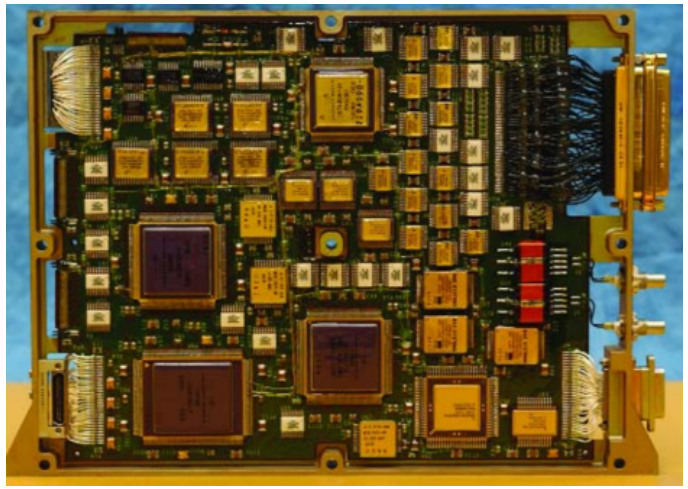
What is the most expensive robotic processor?

Fun Facts

What is the most expensive robotic processor?

RAD 6000

	CompactPCI	Mars Rover	ASIC
Performance	Variable to 26 MIPS	Variable to 22 MIPS	Variable to 35 MIPS
Frequency	Variable to 25 MHz	Variable to 20 MHz	Variable to 33 MHz
Power (maximum)	< 7.5 Watts at 25 MH	20 Watts at 20 MHz	< 10 Watts at 33 MHz
Power (low)	< 3.1 Watts at 5 MHz	5 Watts at 2.5 MHz	2.5 Watts at 4.1 MHz
Memory	8 or 16 MByte SRAM	128 MByte DRAM	8 MByte SRAM
EEPROM/PROM	4 MByte EEPROM	3 MByte EEPROM	Up to 128 Kbyte PROM
Mass (board)	< 0.9 Kg	< 1.2 Kg	< 1.0 Kg
Size	6U Compact PCI	6U VME	IEEE 1101.7 (6" × 9")
Interfaces	1553, UART, PCI, discretes, interrupts	RS232	ASCM module bus, RS422
Support functions	ASIC	FPGA	ASIC

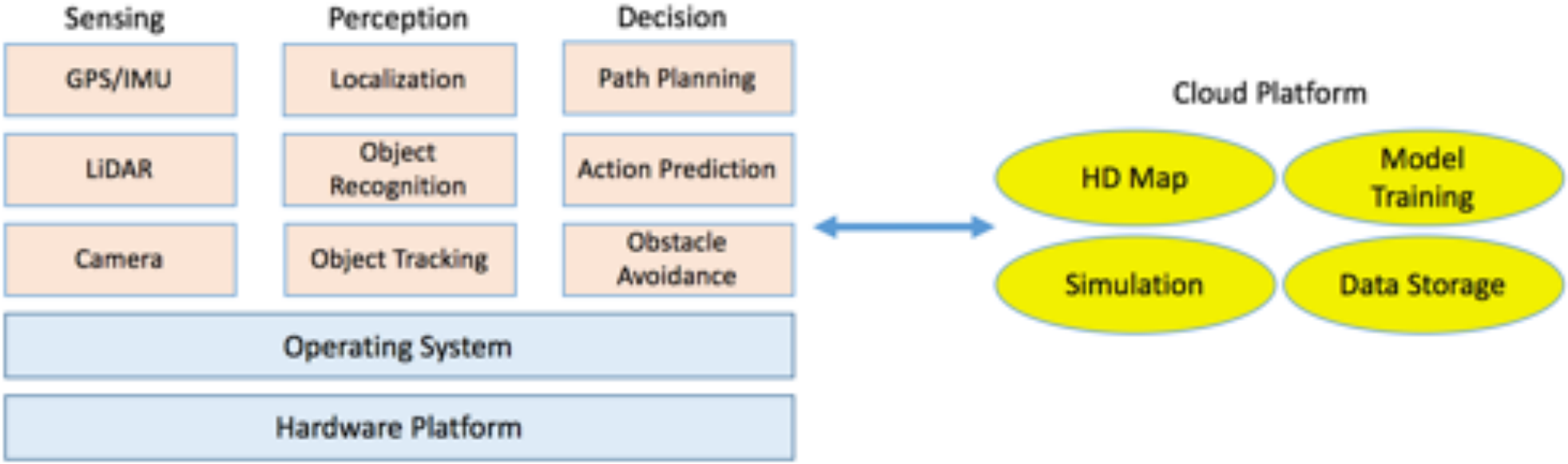


Outline

- Overview of Heavyweight Autonomous Driving
- Overview of Lightweight Autonomous Driving
- Edge Computing: Challenges and Opportunities
- Acceleration Example I: Fully Autonomous Robot Powered by a Cell Phone
- Acceleration Example II: FPGA-based ORB Feature Extraction for Real-Time Visual SLAM
- Acceleration Example III: Bundle Adjustment Acceleration on Embedded FPGAs with Co-observation Optimization
- Potential Class Projects

Autonomous Driving Architecture

Liu, S., Li, L., Tang, J., Wu, S. and Gaudiot, J.L., 2017. Creating autonomous vehicle systems. Synthesis Lectures on Computer Science, 6(1), pp.i-186.

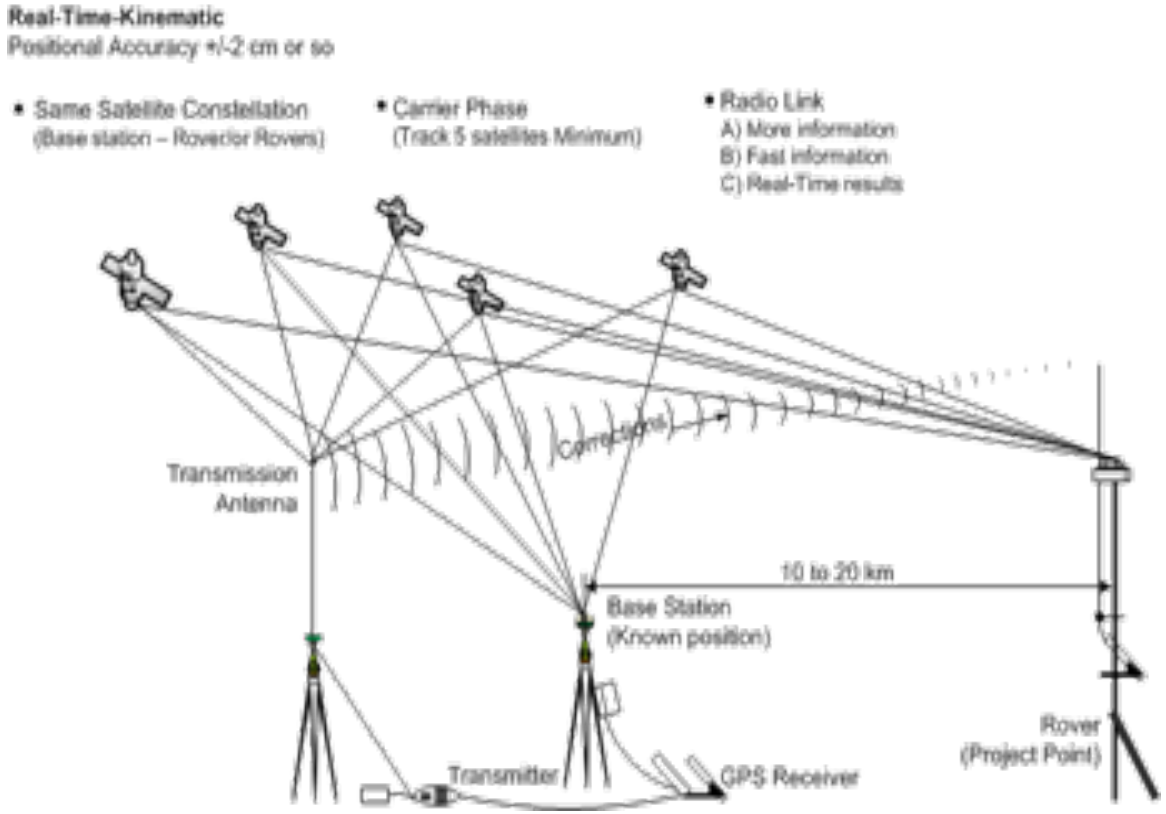


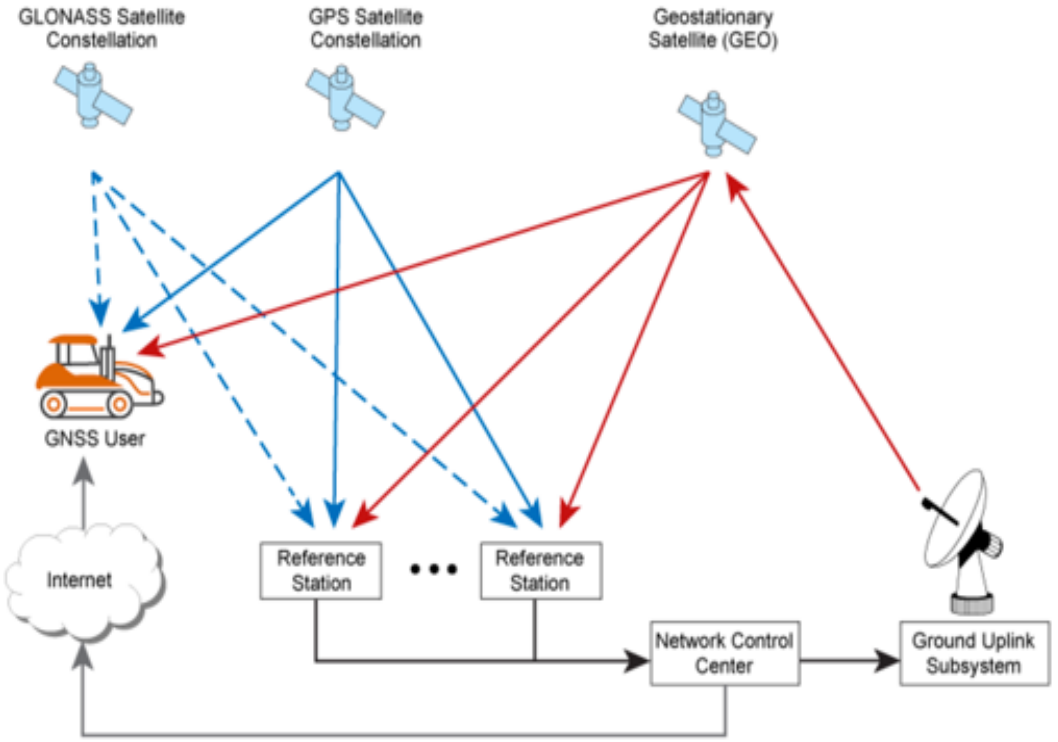
Localization: GNSS/INS

Contributing Source	Error Range
Satellite Clocks	±2 m
Orbit Errors	±2.5 m
Inospheric Delays	±5 m
Tropospheric Delays	±0.5 m
Receiver Noise	±0.3 m
Multipath	±1 m

Building Blocks:

- *RTK GNSS*
- *PPP*



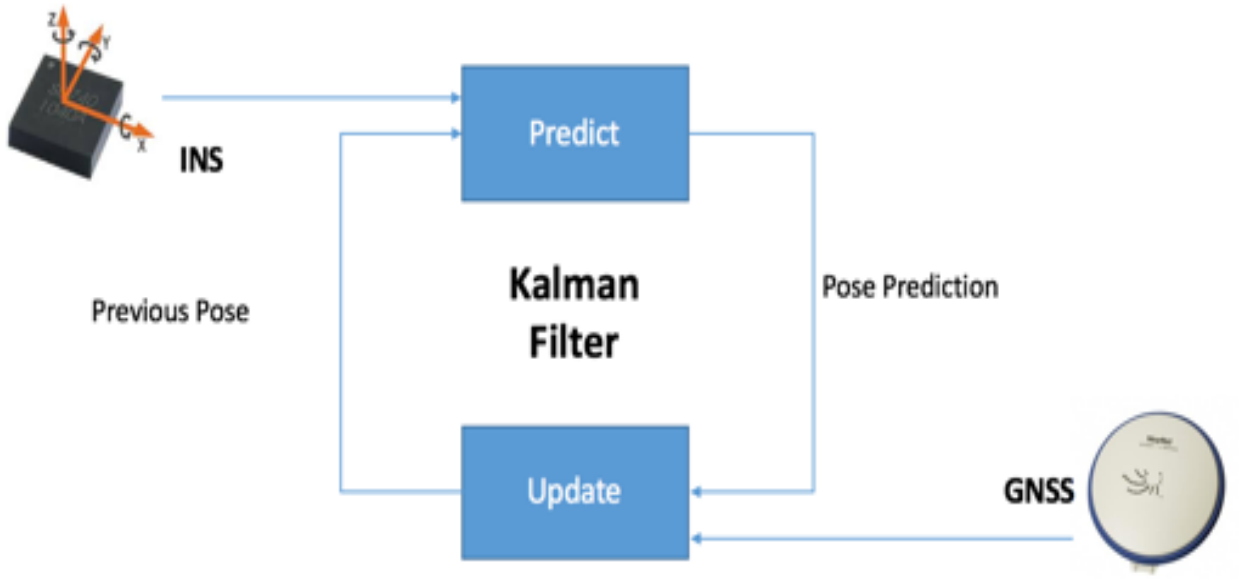


Localization: GNSS/INS

- **GNSS**：accurate but low frequency
- **INS**：high frequency but inaccurate
- **Kalman Filter:** get the best from both
- **Question:** is GPS/IMU enough?

Building Blocks:

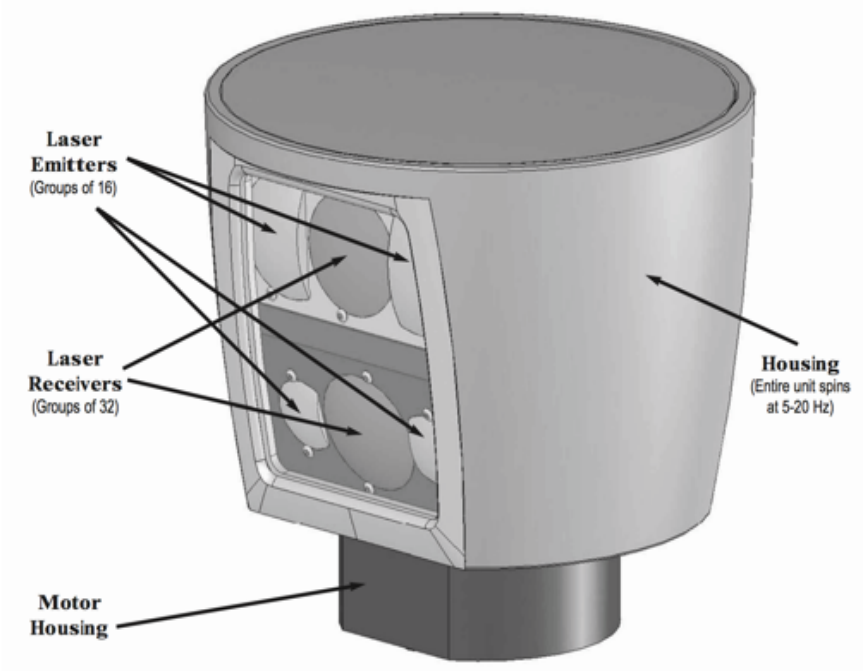
- *Kalman filters*

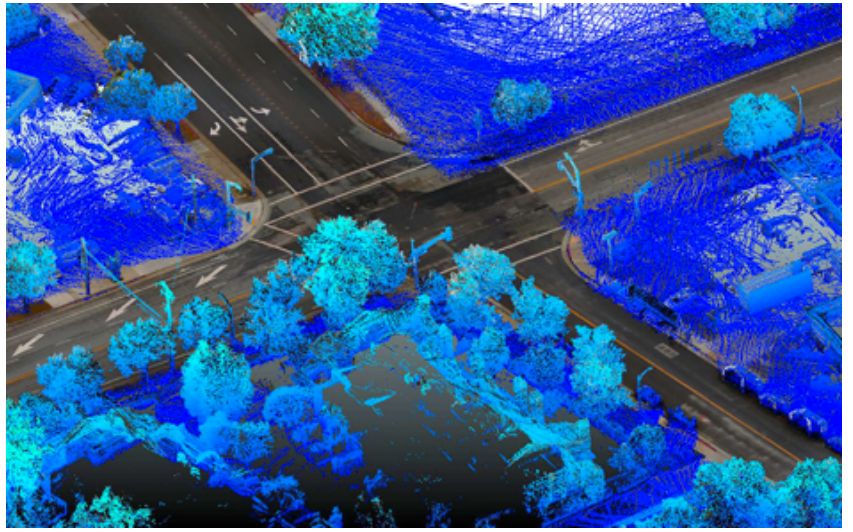


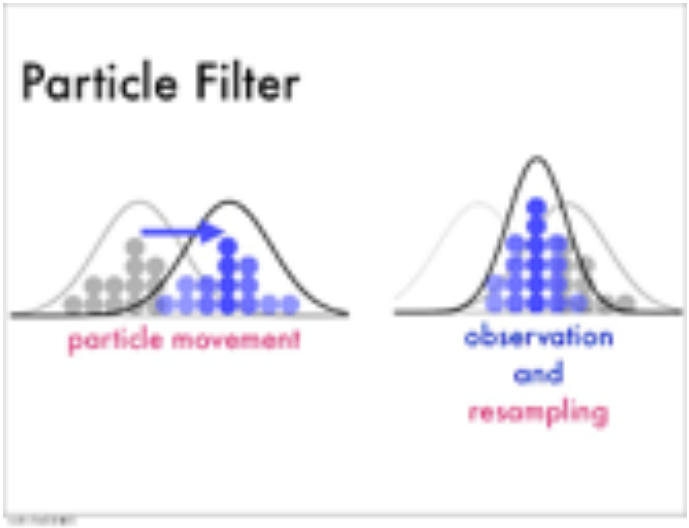
Localization: LiDAR and HD Map

Building Blocks:

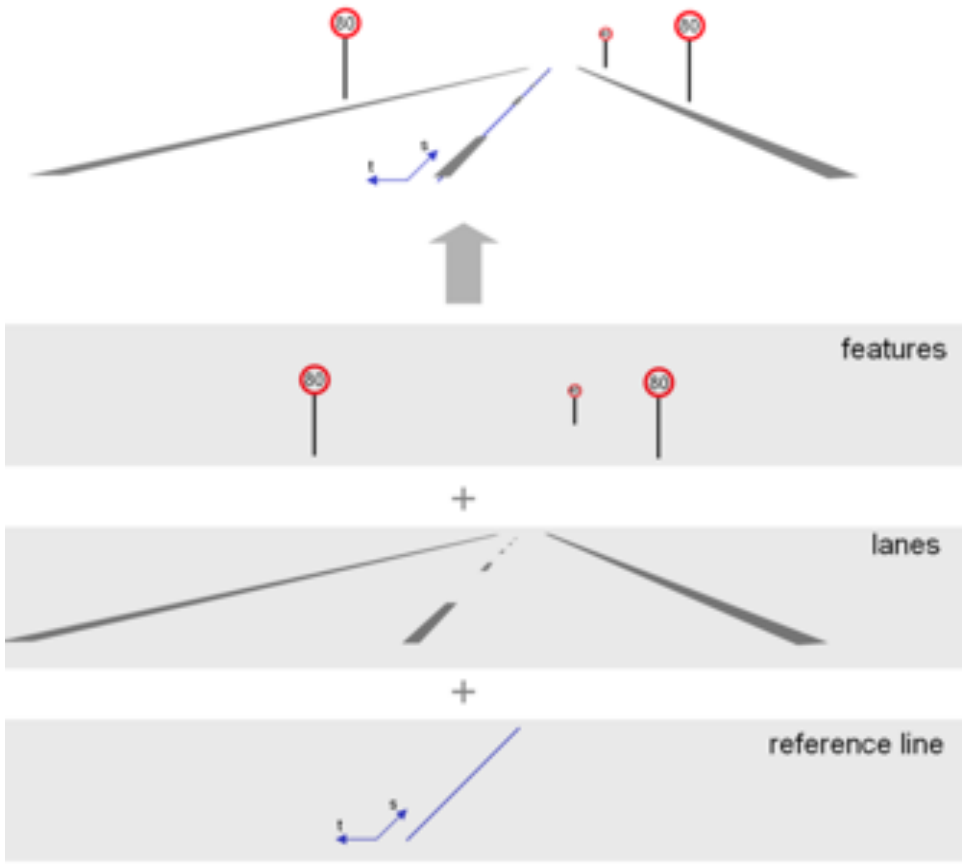
- *Particle Filters*
- *Iterative Closest Point (ICP)*







Localization: LiDAR and HD Map



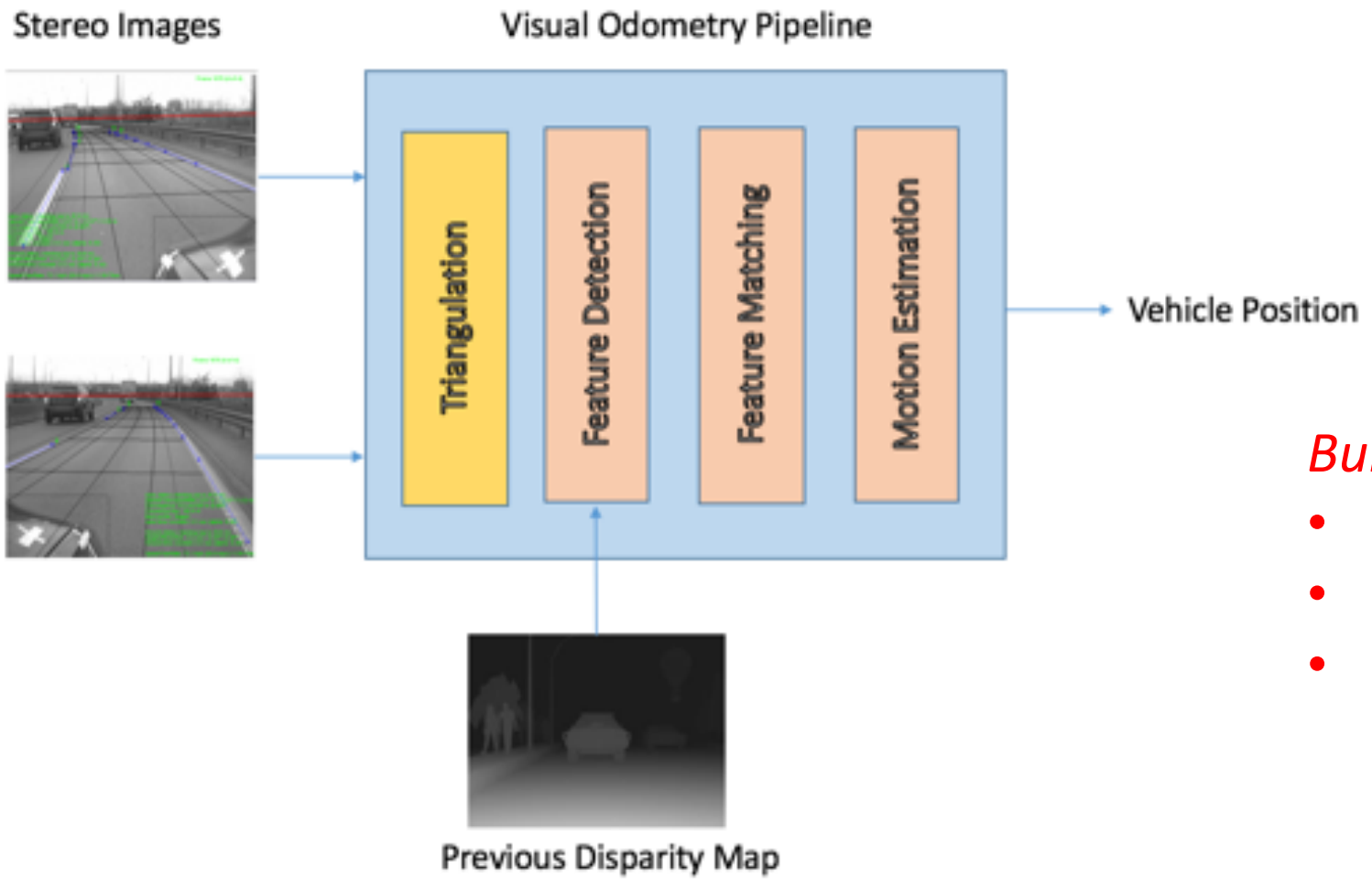
Layered Map:

- Foundation layer is a 2D grid map with 5 cm by 5 cm resolution
- Road reference line is then added
- Next layer is the lane information
- Other semantic feature are then added

Requirements:

- Fresh
- Precise
- Integration with the client systems

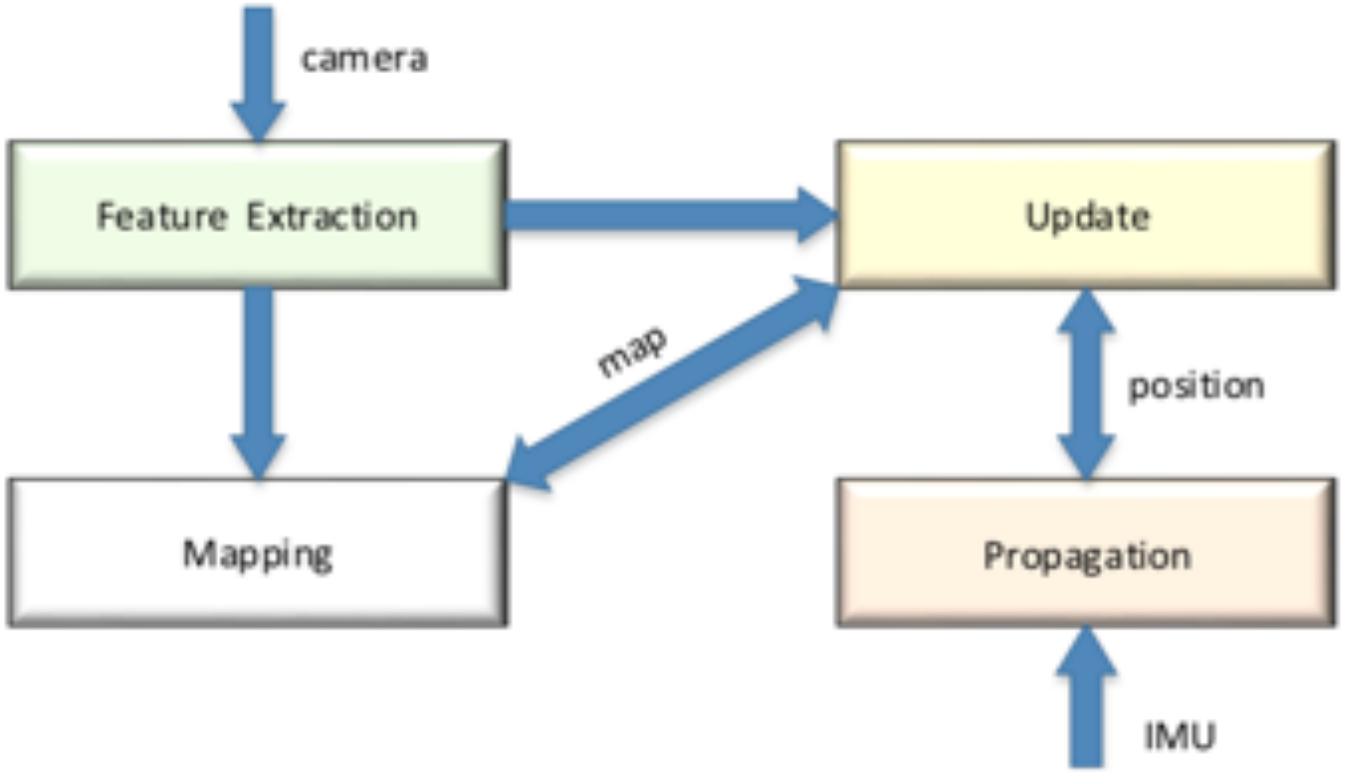
Localization: VSLAM



Building Blocks:

- *Stereo visual odometry*
- *Mono visual odometry*
- *Visual inertial odometry*

Localization: VSLAM



Building Blocks:

- *Feature Extraction*
- *Extended Kalman Filter (EKF)*
- *Bundle Adjustment (BA)*

Localization: Sensor Fusion

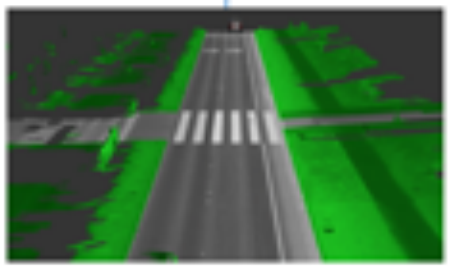
~ $80000

Propagation

Pose Prediction

Sensor Fusion

Pose



~ tens of millions USD

Computer Vision: Datasets

Geiger, Andreas ; Lenz, Philip ; Stiller, Christoph ; Urtasun, Raquel: Vision meets Robotics: The KITTI Dataset.
International Journal of Robotics Research (IJRR) 32 (2013), pp. 1229–1235

Vision meets Robotics: The KITTI Dataset

Andreas Geiger, Philip Lenz, Christoph Stiller and Raquel Urtasun

***Abstract*—We present a novel dataset captured from a VW station wagon for use in mobile robotics and autonomous driving research. In total, we recorded 6 hours of traffic scenarios at 10-100 Hz using a variety of sensor modalities such as high-resolution color and grayscale stereo cameras, a Velodyne 3D laser scanner and a high-precision GPS/IMU inertial navigation system. The scenarios are diverse, capturing real-world traffic situations and range from freeways over rural areas to inner-city scenes with many static and dynamic objects. Our data is calibrated, synchronized and timestamped, and we provide the rectified and raw image sequences. Our dataset also contains object labels in the form of 3D tracklets and we provide online benchmarks for stereo, optical flow, object detection and other tasks. This paper describes our recording platform, the data format and the utilities that we provide.**

***Index Terms*—dataset, autonomous driving, mobile robotics, field robotics, computer vision, cameras, laser, GPS, benchmarks, stereo, optical flow, SLAM, object detection, tracking, KITTI**

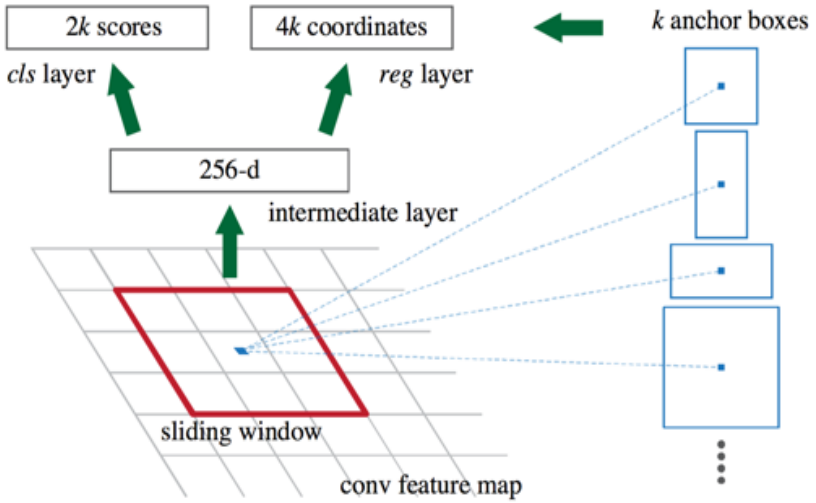
I. Introduction

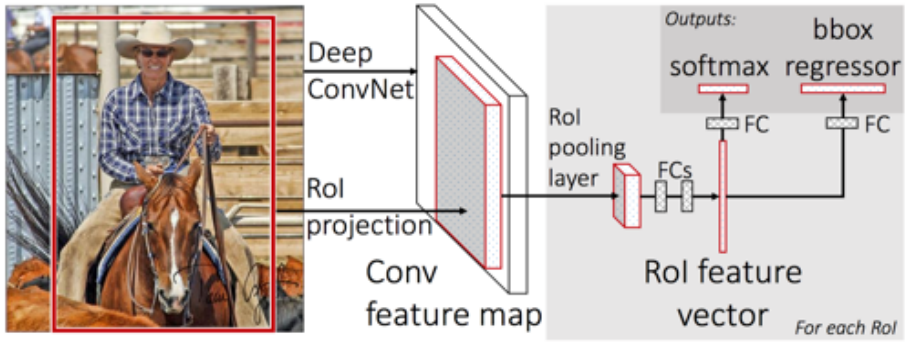
The KITTI dataset has been recorded from a moving platform (Fig. 1) while driving in and around Karlsruhe, Germany (Fig. 2). It includes camera images, laser scans, high-precision GPS measurements and IMU accelerations from a combined

Fig. 1. **Recording Platform.** Our VW Passat station wagon is equipped with four video cameras (two color and two grayscale cameras), a rotating 3D laser scanner and a combined GPS/IMU inertial navigation system.

- 1 × OXTS RT3003 inertial and GPS navigation system, 6 axis, 100 Hz, L1/L2 RTK, resolution: 0.02m / 0.1°

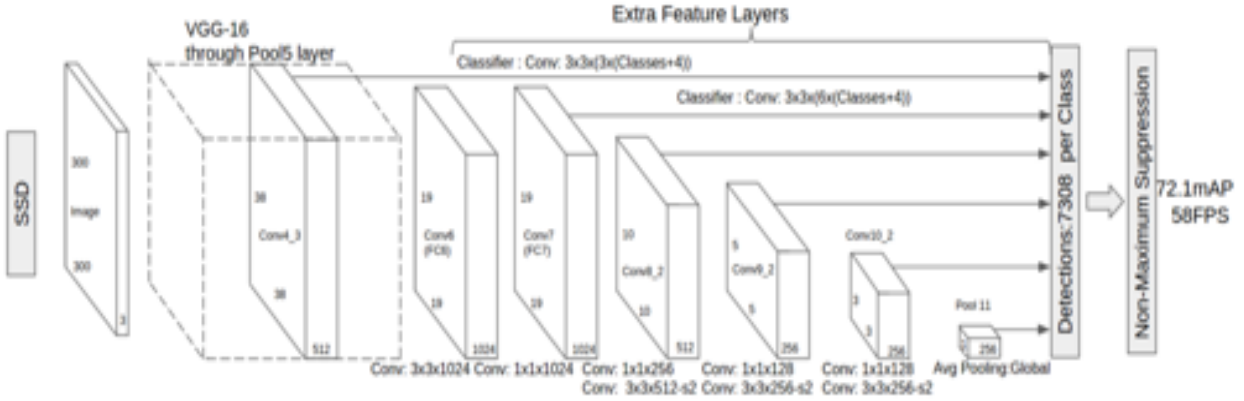
Detection: Mask R-CNN and SSD



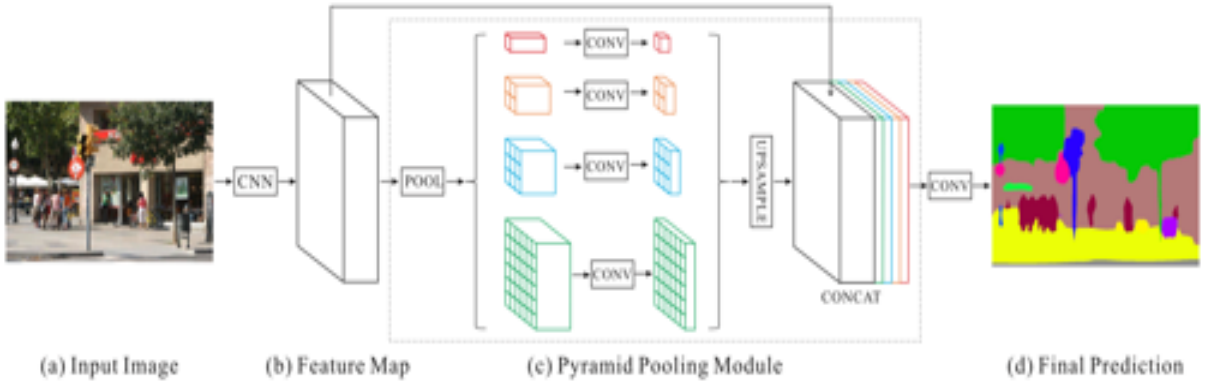


Building Blocks:

- *Mask R-CNN*
- *SSD*

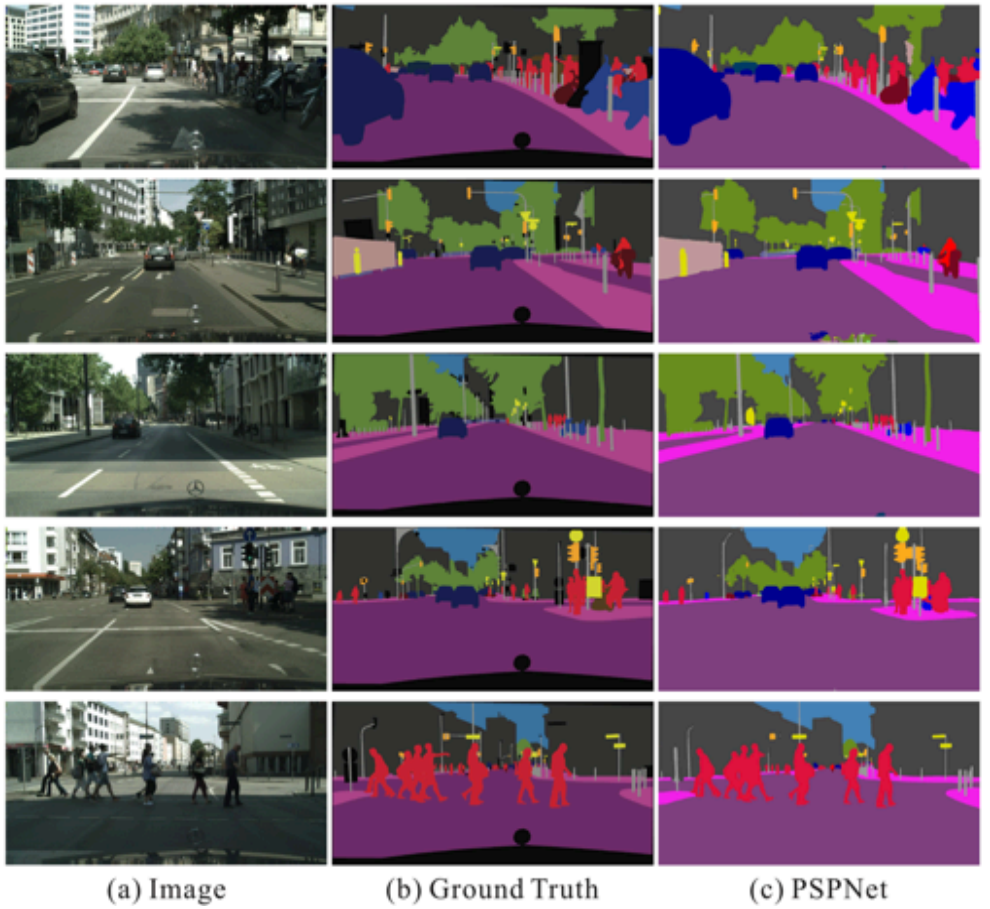


Semantic Segmentation

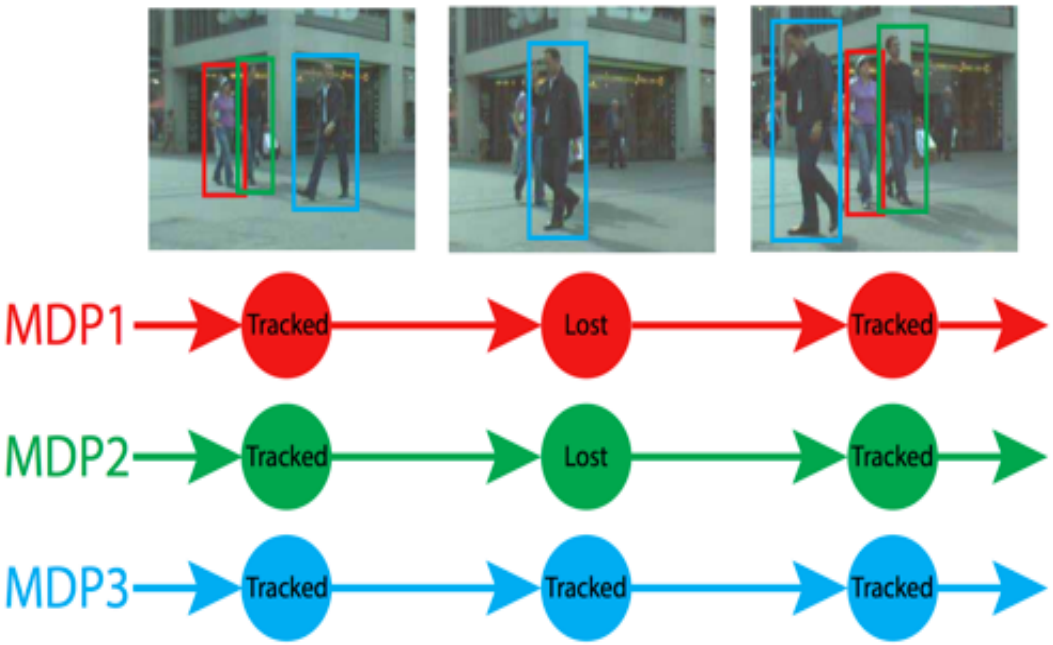


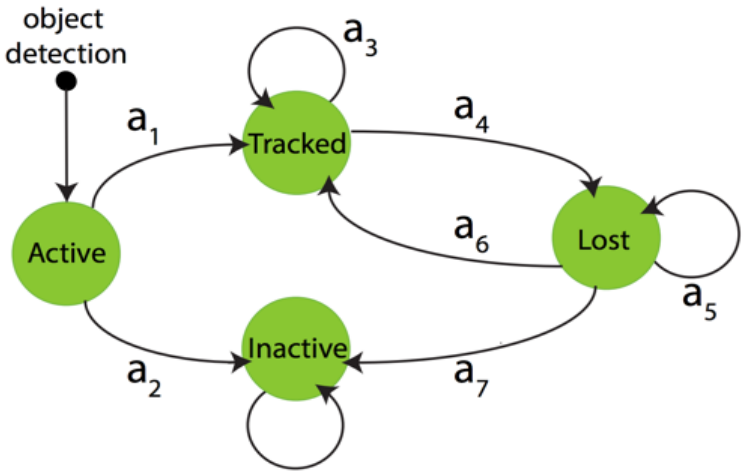
Building Blocks:

- *PSPNet*



Object Tracking

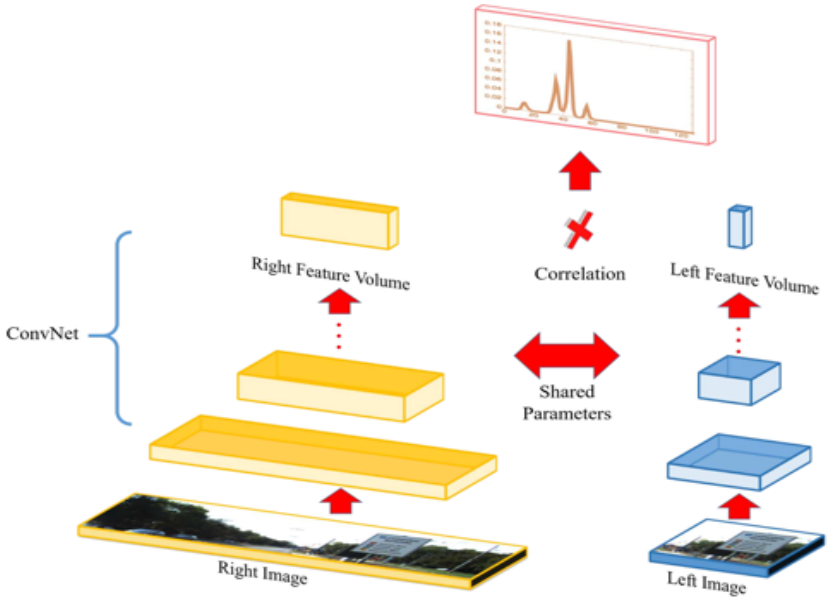




Building Blocks:

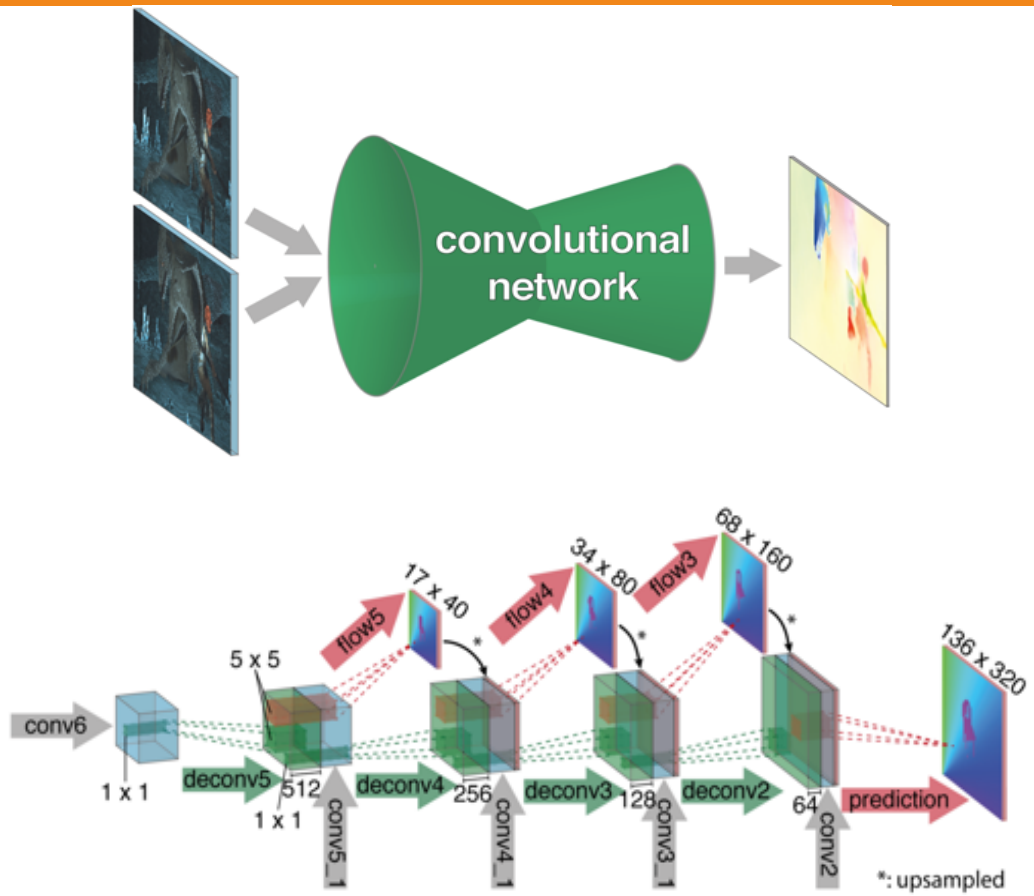
- *MDP*
- *POMDP*

Stereo and Optical Flow



Building Blocks:

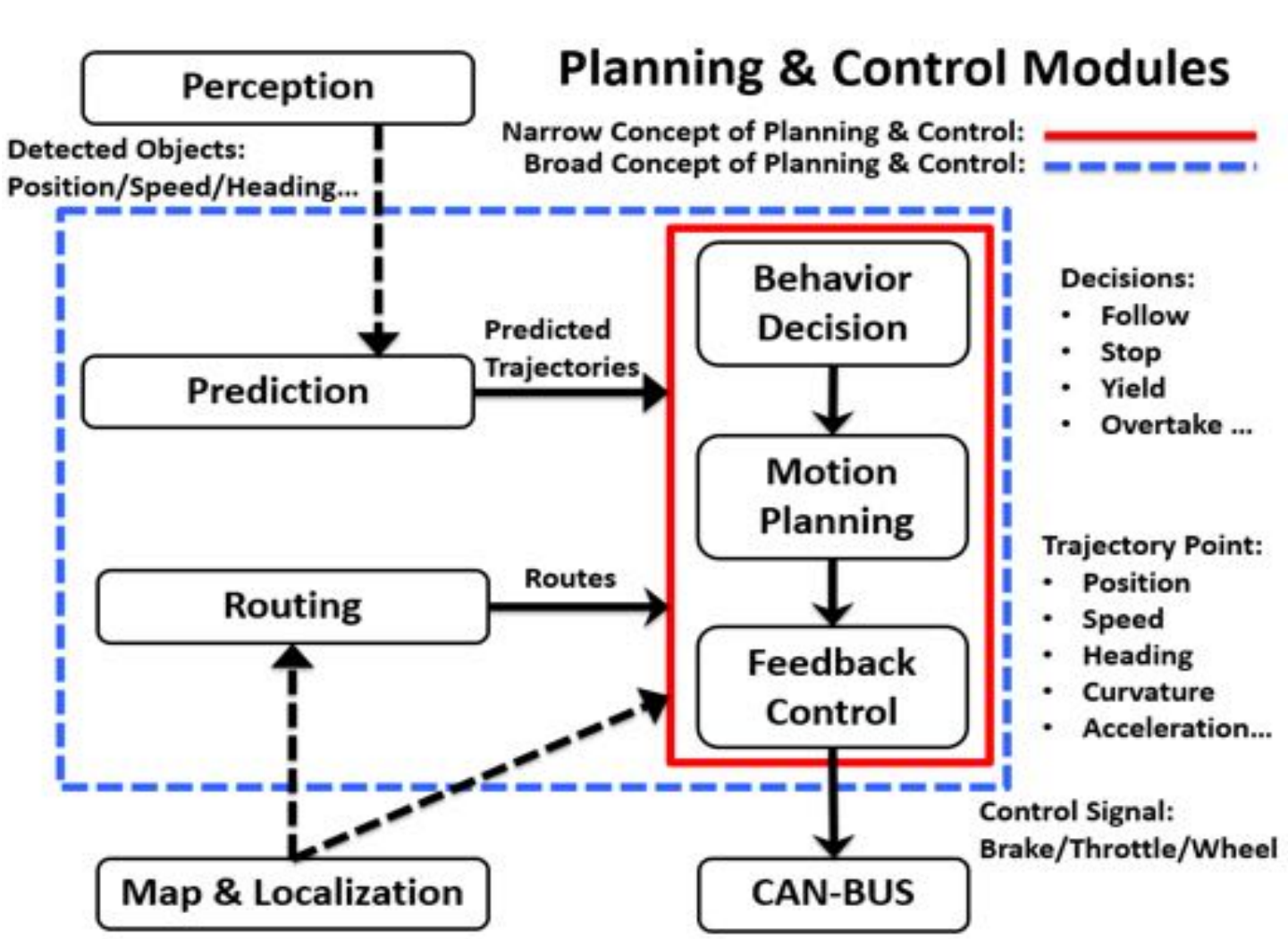
- *Content CNN*
- *ELAS*

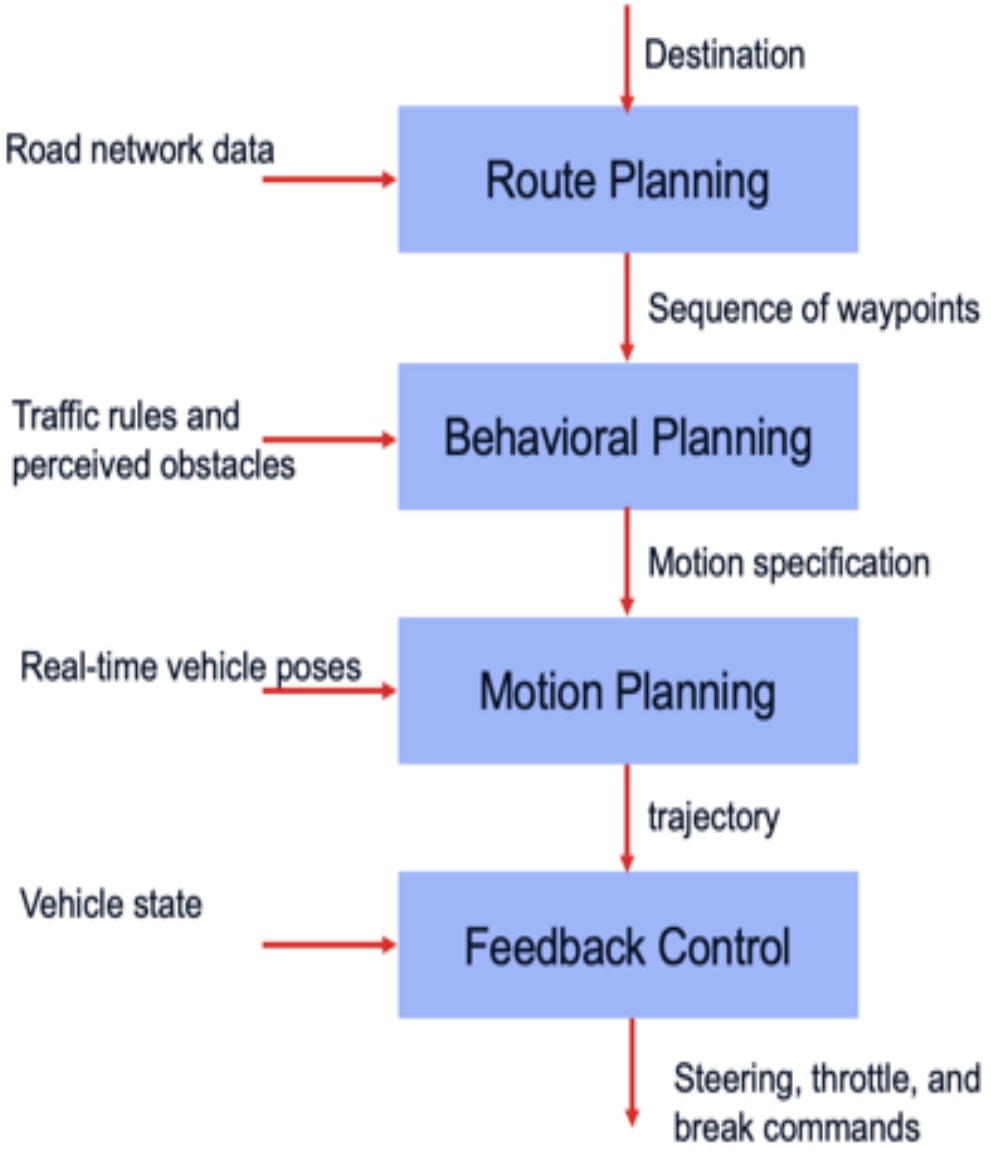


Building Blocks:

- *Optical Flow*
- *FlowNet*

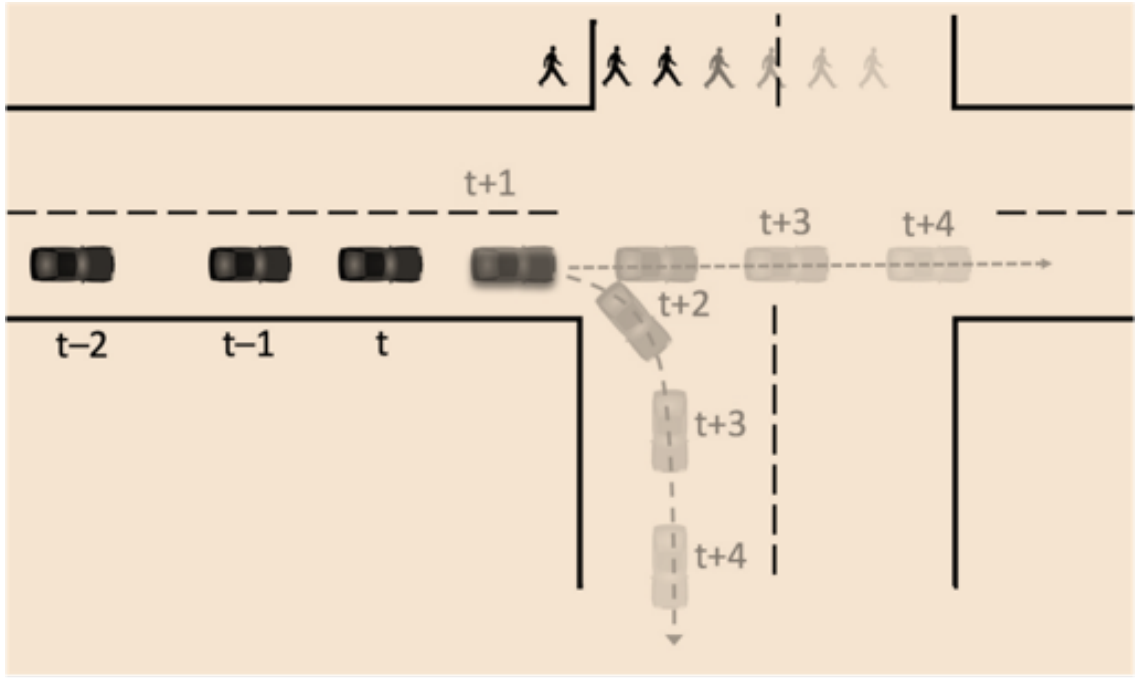
Planning and Control Architecture

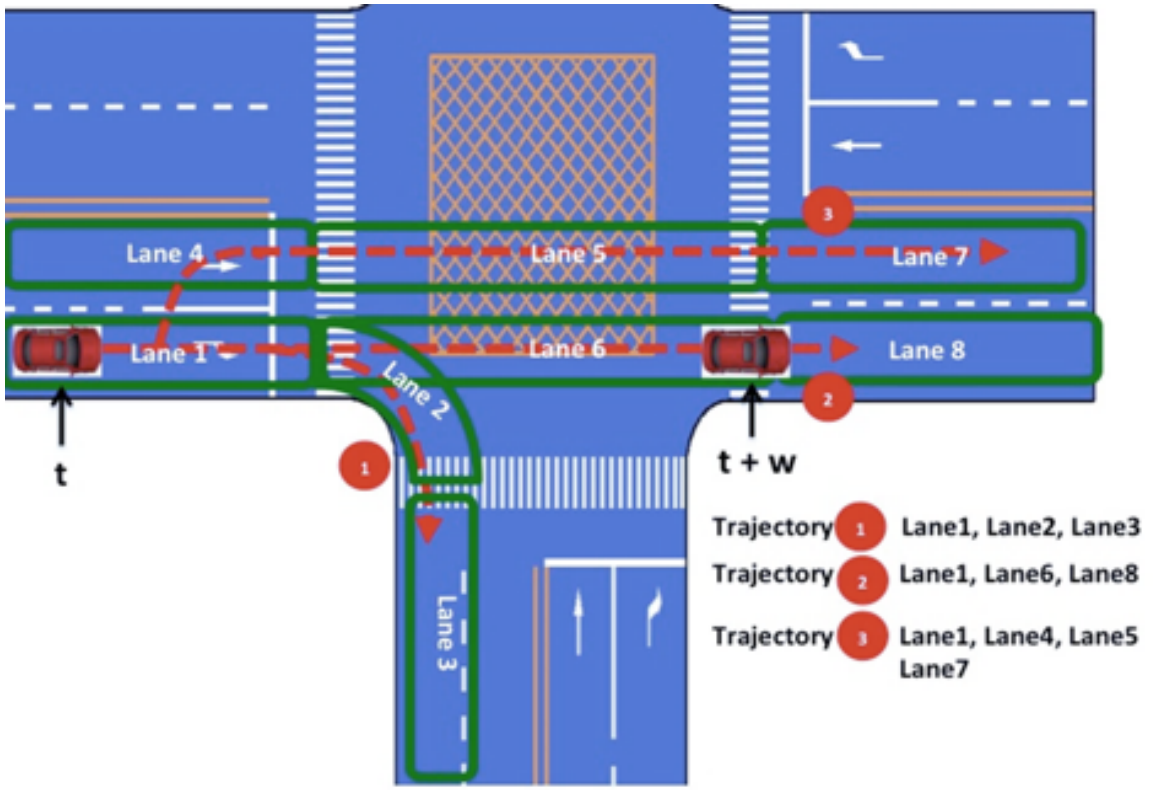




Traffic Prediction

- *Classification problem* for categorical road object behaviors
- *Regression problem* for generating the predicted path with speed and time info



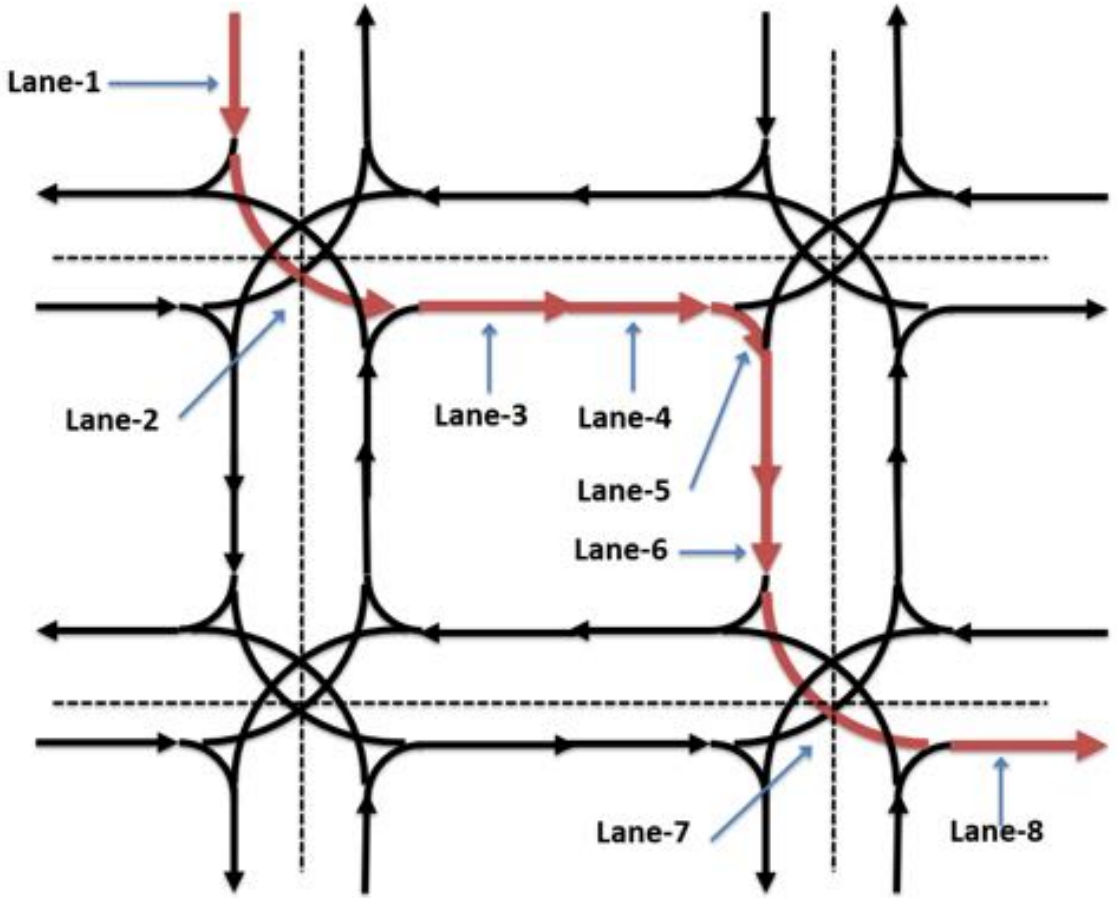


Building Blocks:

- *POMDP*

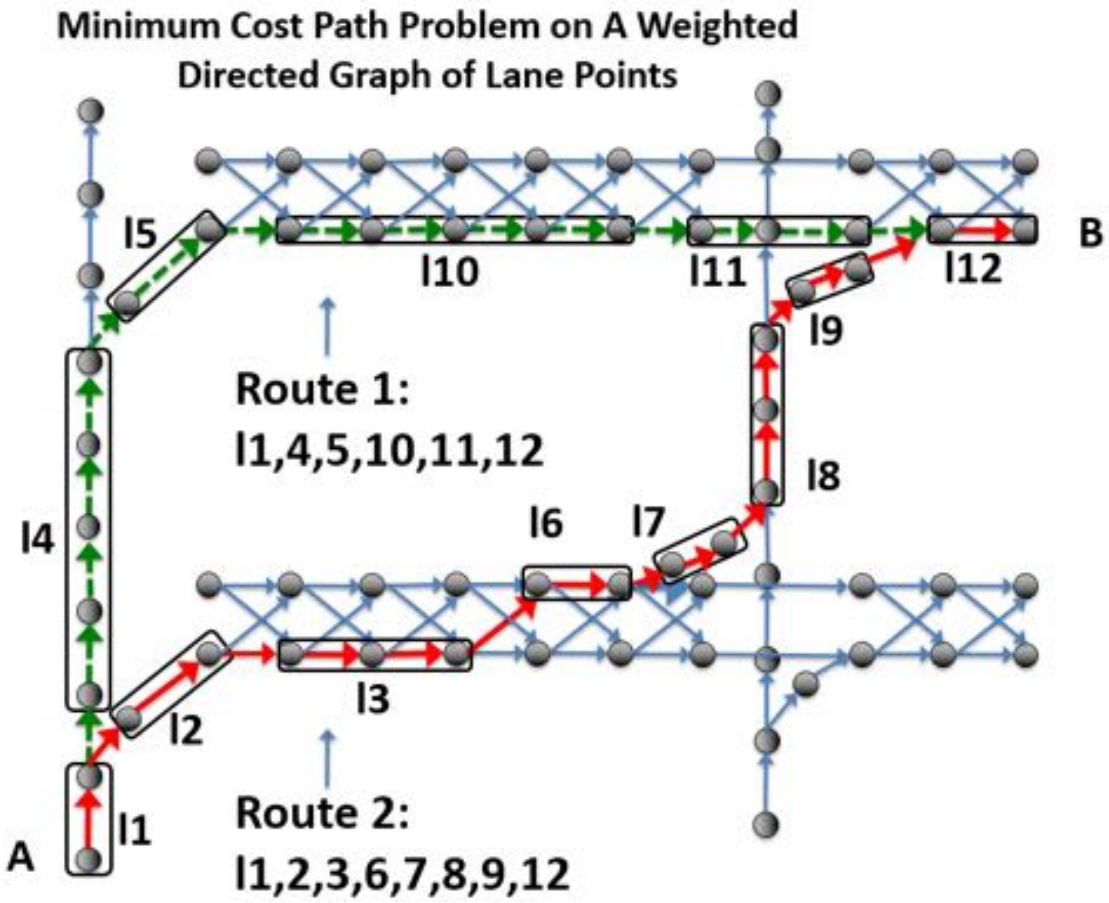
Lane-Level Routing

- Modeled as weighted directed graphs



Building Blocks:

- *Dijkstra*
- *A**

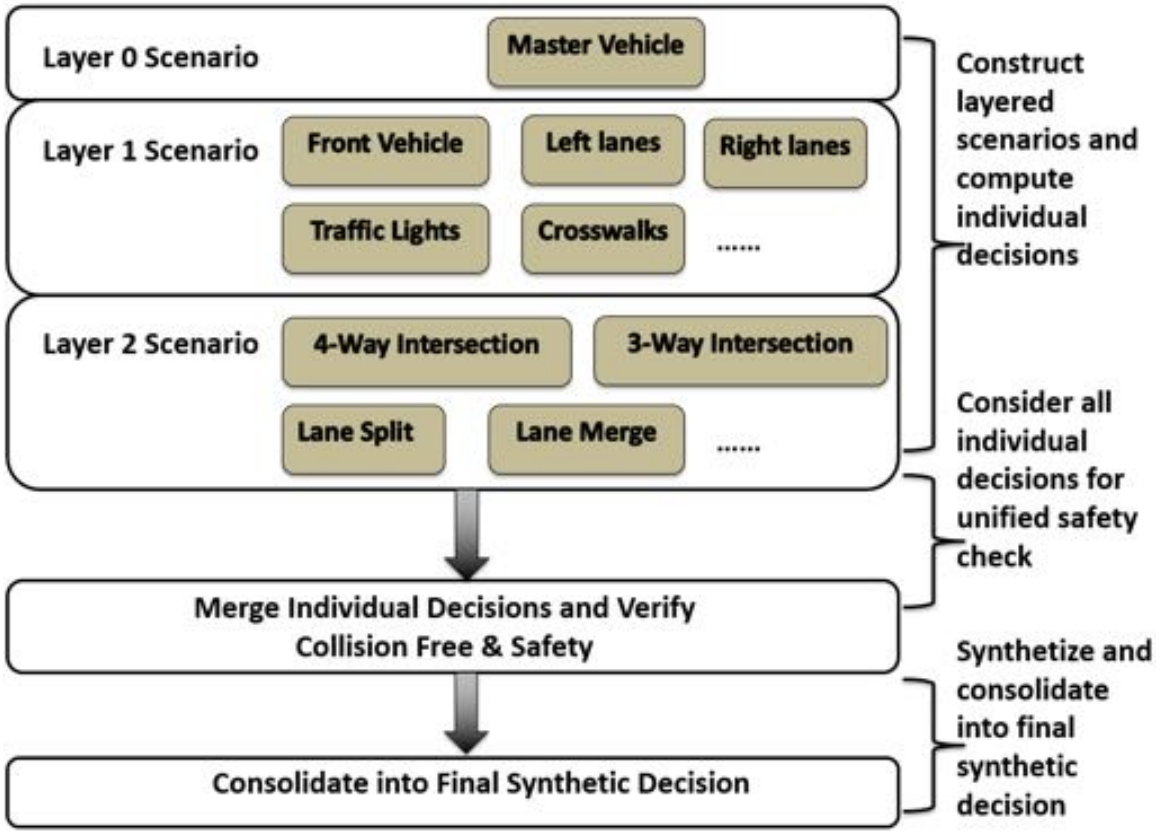


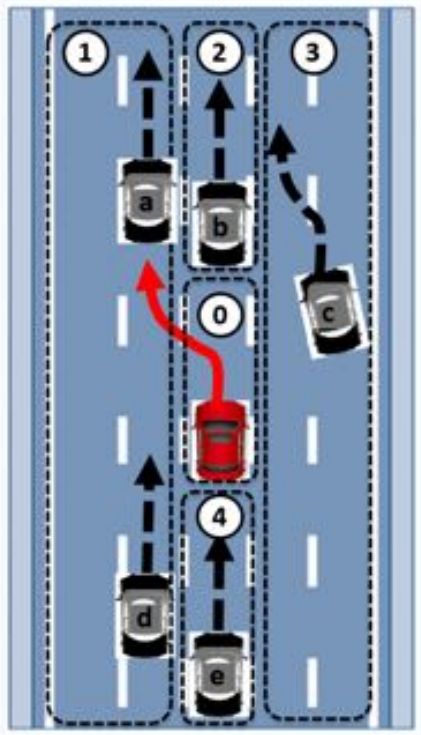
Behavioral Decisions

- Ruled-based “divide-and-conquer” approach: layered scenarios
- Synthetic decision and individual decisions

Building Blocks:

- *MDP*





Synthetic Decision:
Switch lane from current lane to the left lane: yield vehicle a, overtake vehicle d, and attention to vehicle b at current lane

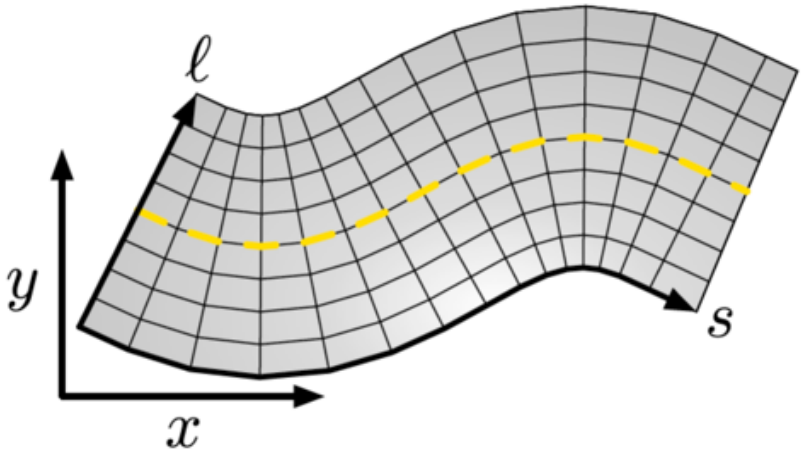
Scenarios and Individua Decisions:
0.Master Vehicle
1.Left Lane(s) : Overtake d and yield a
2.Front Vehicle(s): Attention b
3.Right Lane(s): Ignore c
4.Rear Vehicles(s): Ignore e

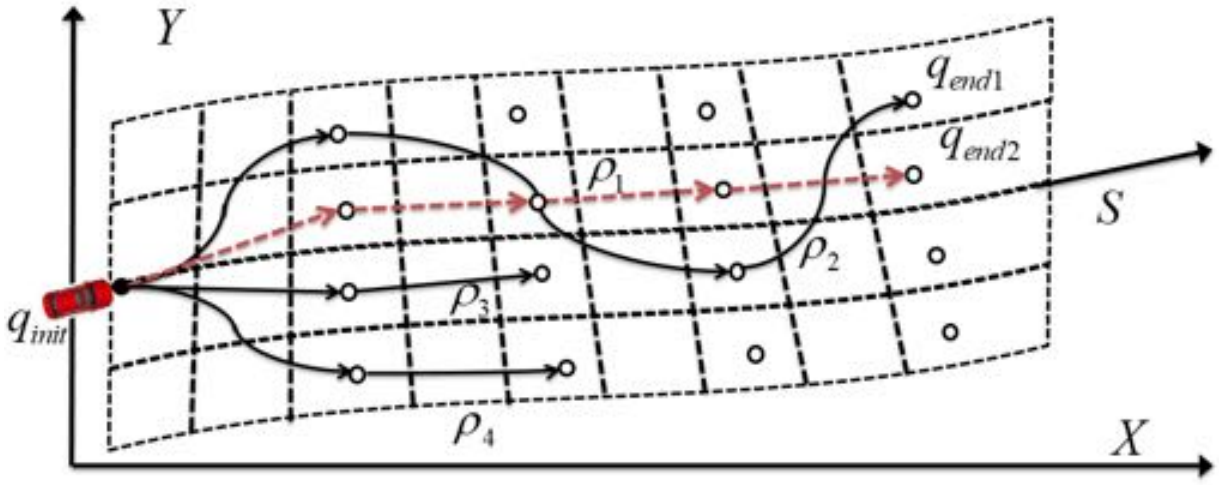
Motion Planning

- Vehicle model, road model, and SL-coordination system
- Path planning and speed planning
  - Path planning: dynamic programming to minimize cost
  - Speed planning: ST-graph

Building Blocks:

- *DP and QP*
- *RRT**
- *E2E methods*



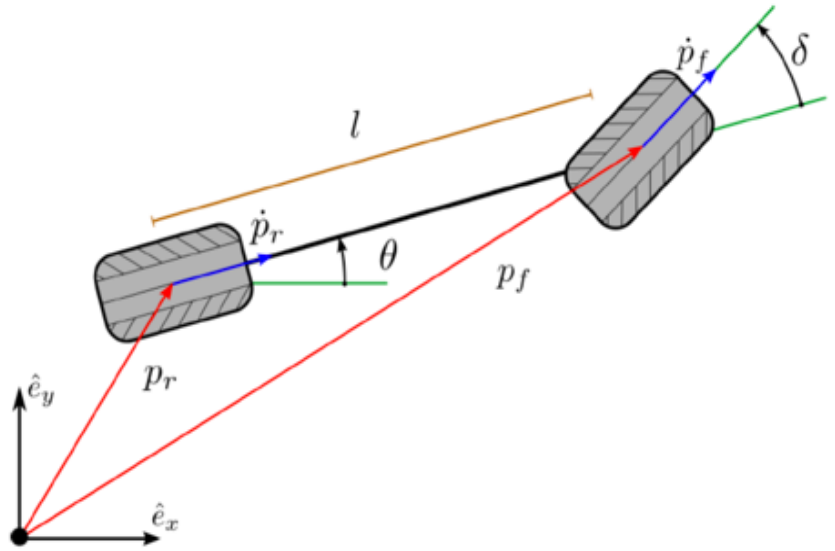


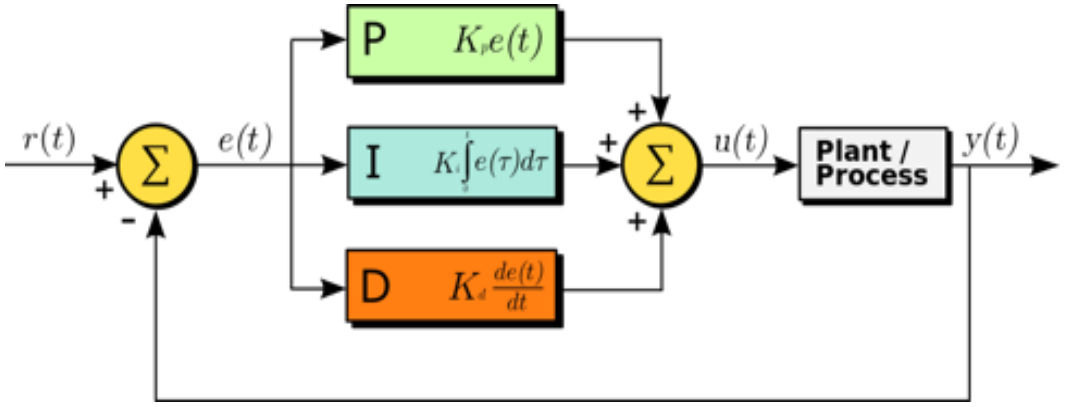
Feedback Control

- Bicycle model: model vehicles as rigid bodies with front and rear wheels
- PID control: proportional-integral-derivative controller

Building Blocks:

- *PID*
- *MPC*

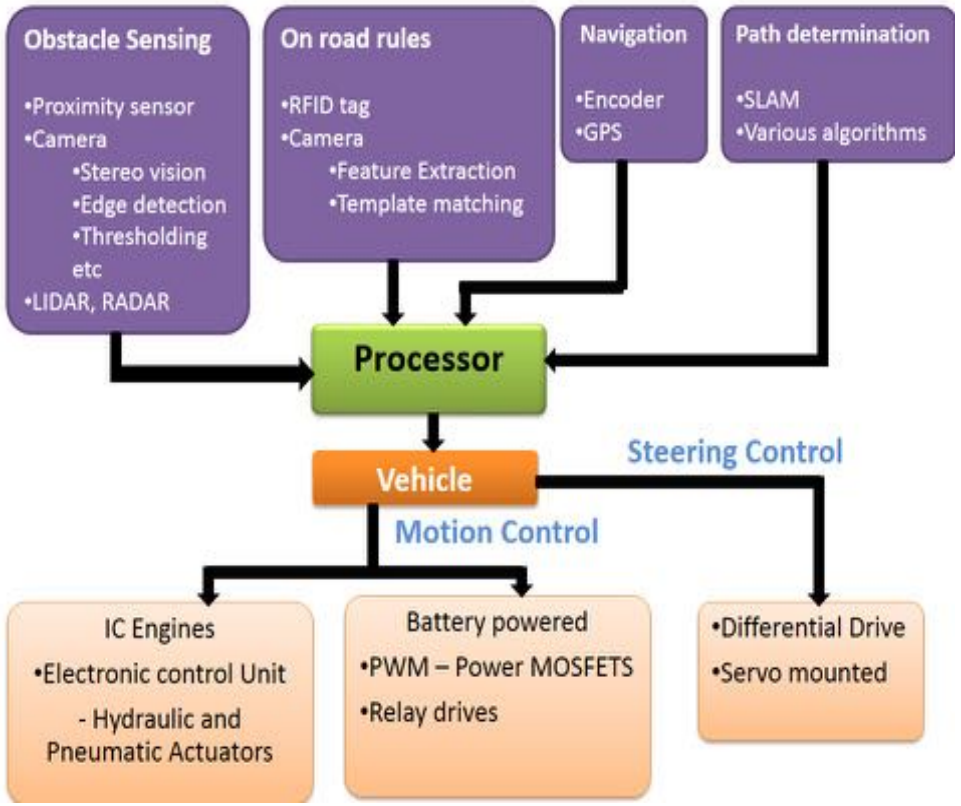




Middleware A.K.A. "*Operating System*"

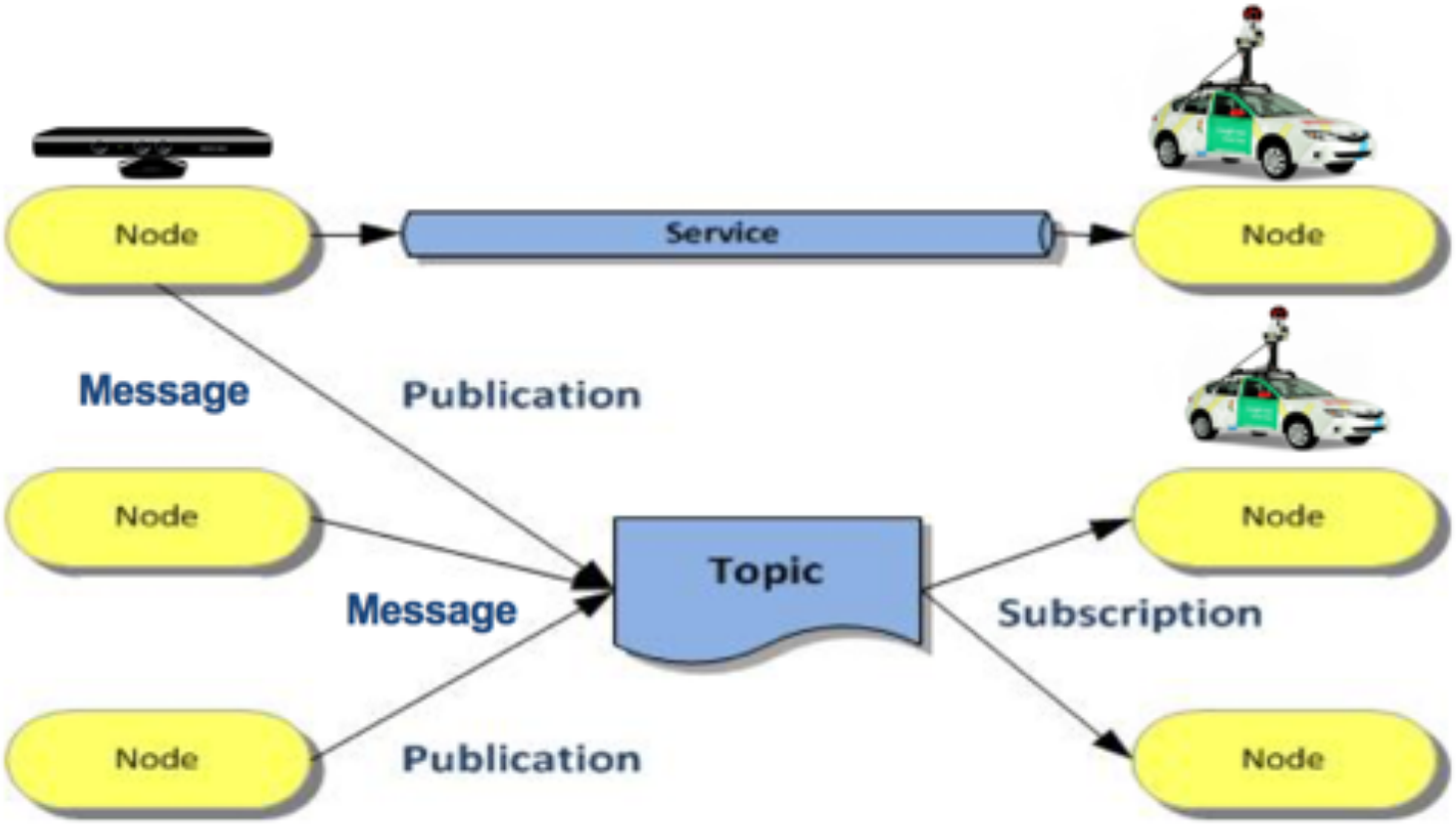
It takes a robust operating system to handle the complex interactions between different components

e.g. 60 FPS requirements for camera, leaving 16 ms for each frame, huge challenge when the amount of data becomes overwhelming



Middleware: ROS

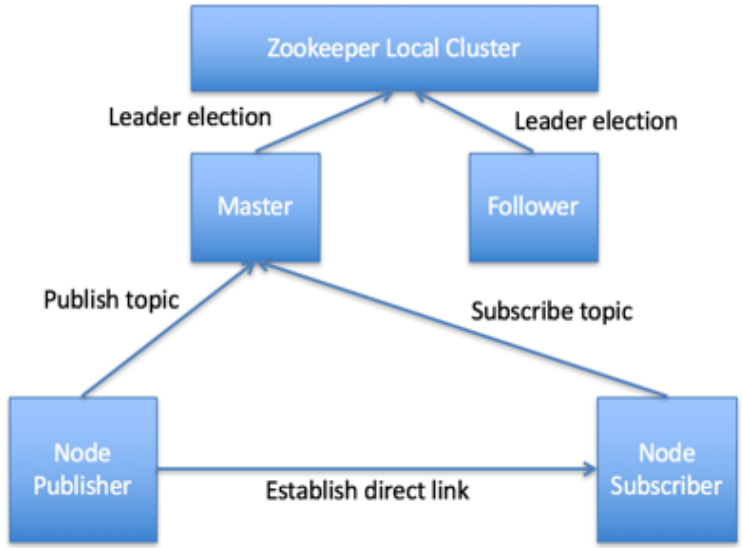
ROS provides a mechanism for the components to interact with each other

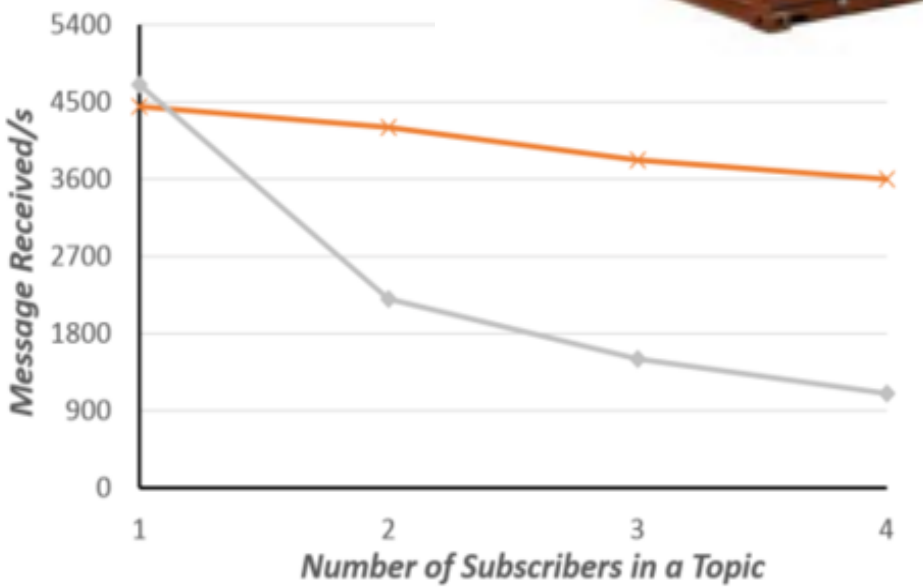


Middleware: ROS

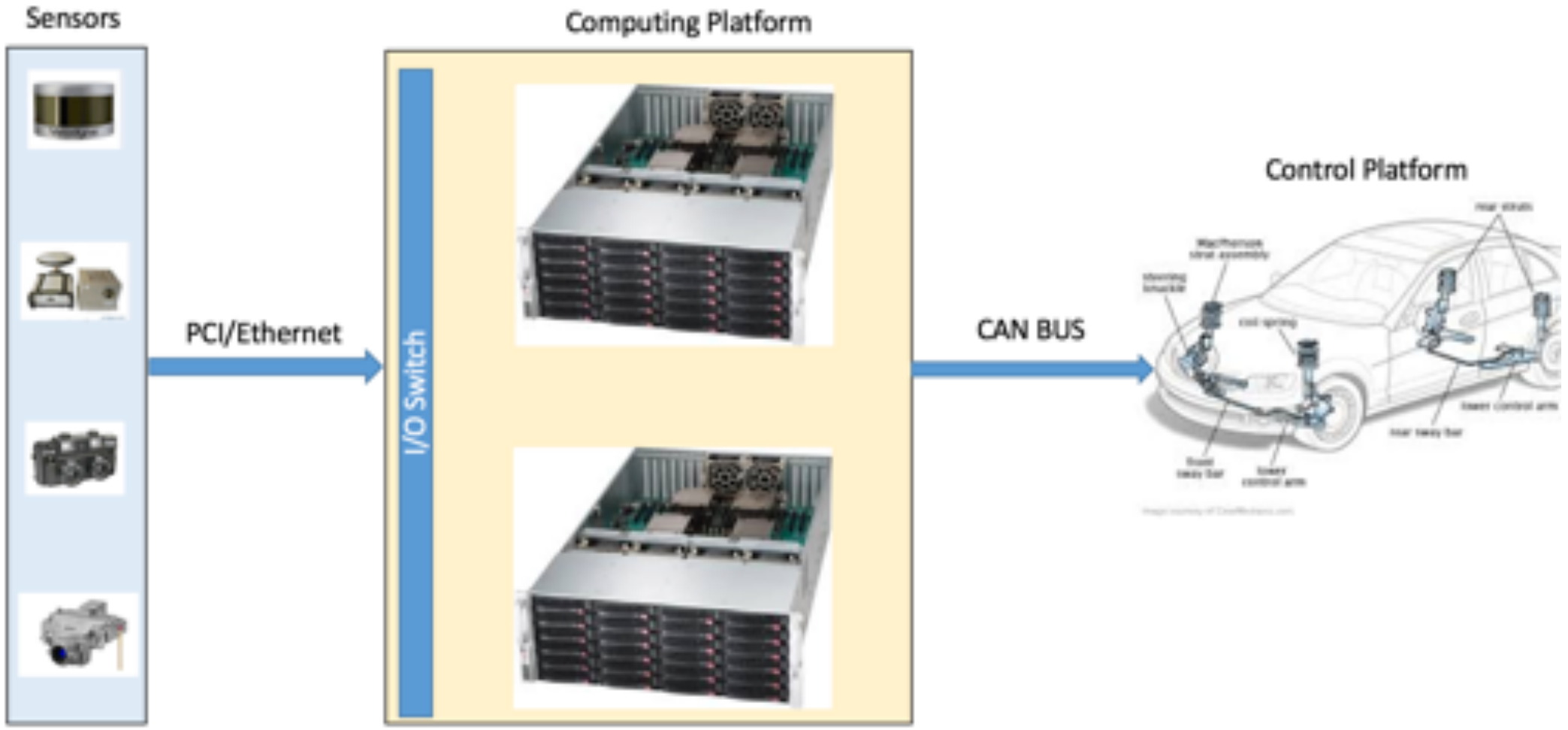
ROS Problems

- Not efficient: communication becomes the bottleneck
- Not secure: no protection mechanism
- Not scalable: central master node
- Not reliable: easily crashed



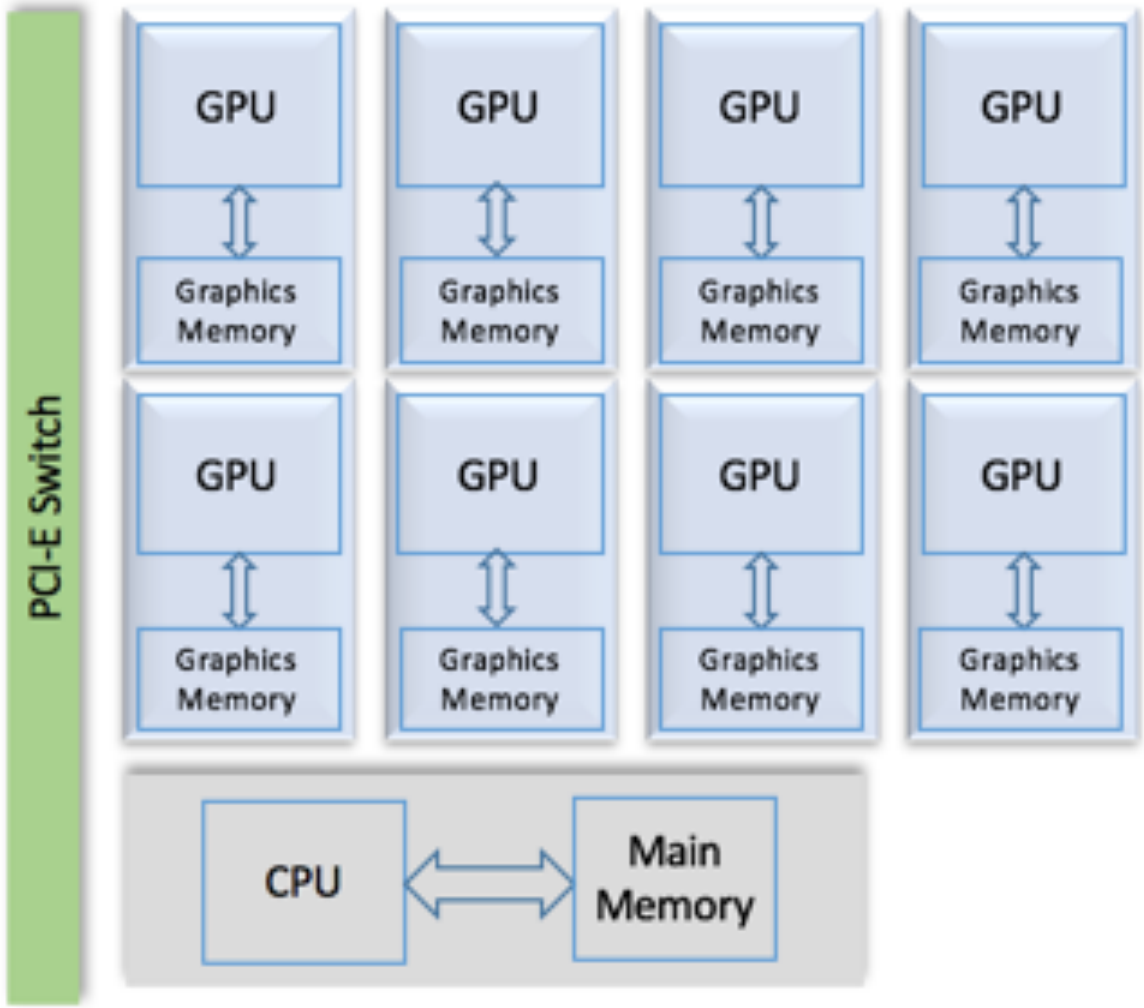


Hardware Platform



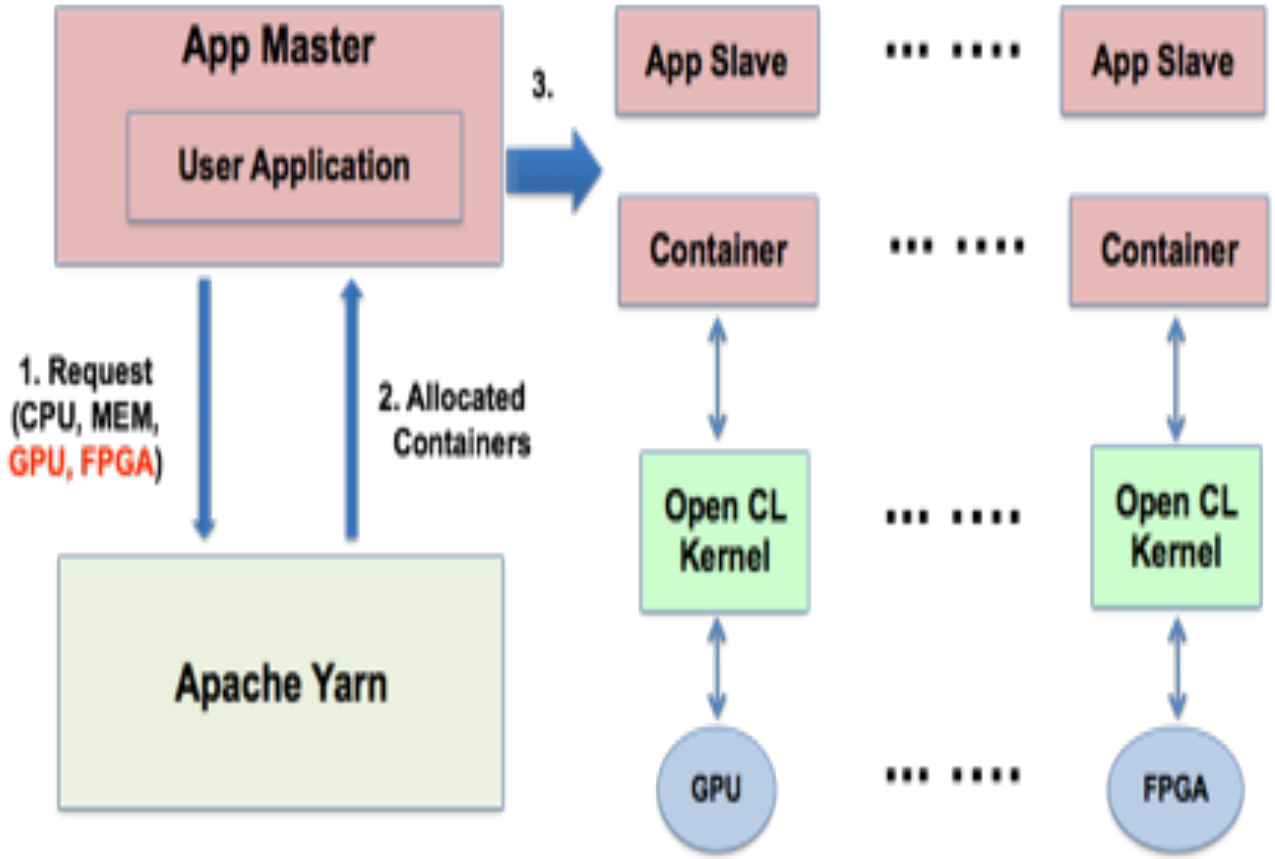
Hardware Platform

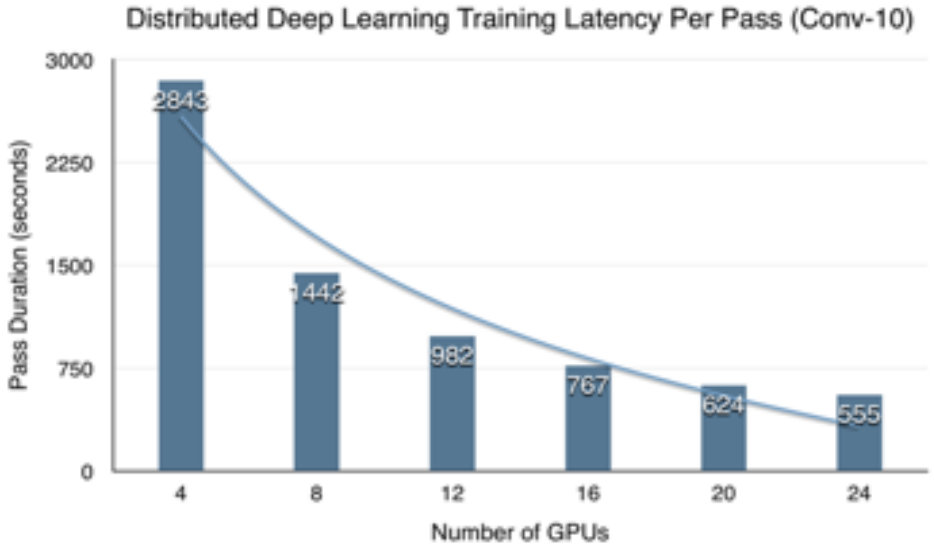
- High Performance
  - CPU + 8 ~ 16 GPUs
  - 60 TOPS/s
- High-Power Consumption
  - 3000 W at peak
- High Cost
  - $20000 ~ $30000
- Heat Dissipation
  - Special fan design needed



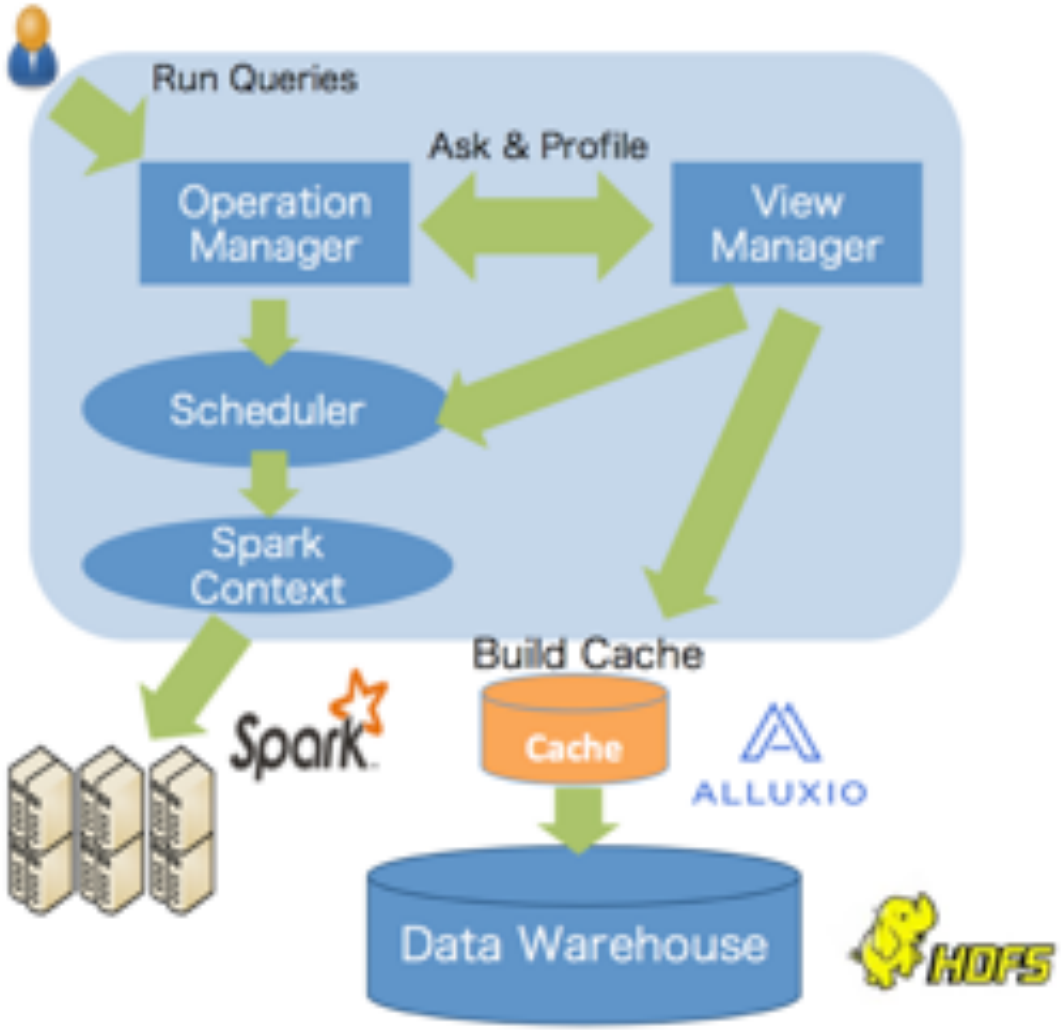
Cloud Infrastructure: Computing

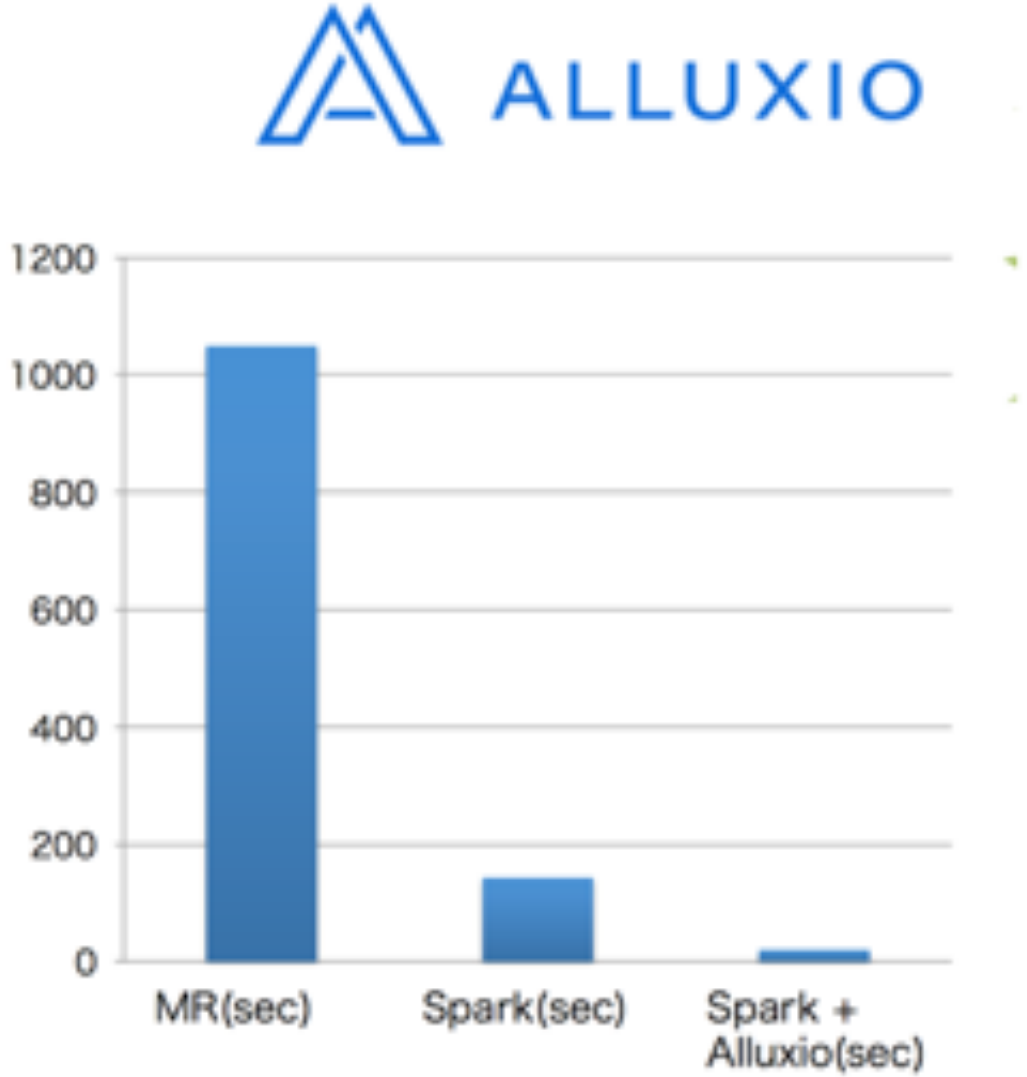
Liu, S., Tang, J., Wang, C., Wang, Q. and Gaudiot, J.L., 2017. A unified cloud platform for autonomous driving. Computer, 50(12), pp.42-49.



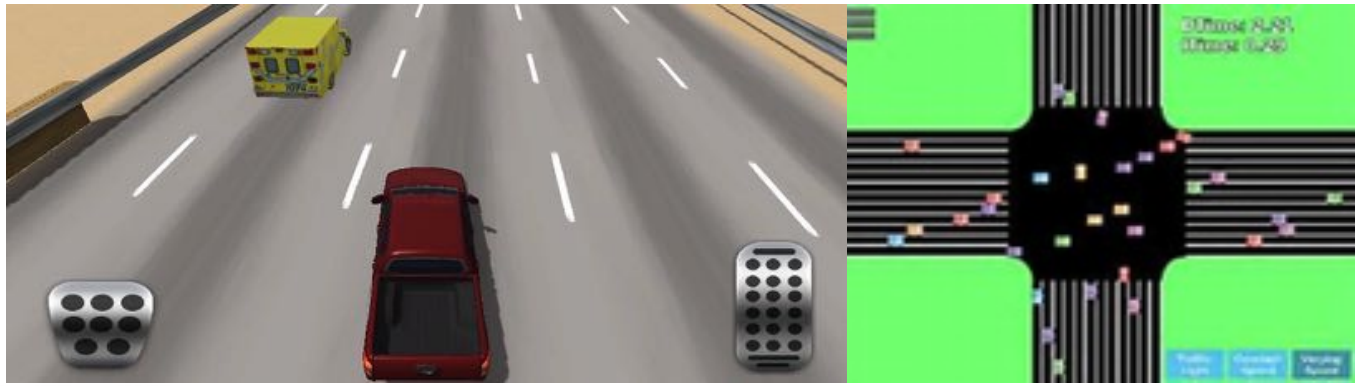


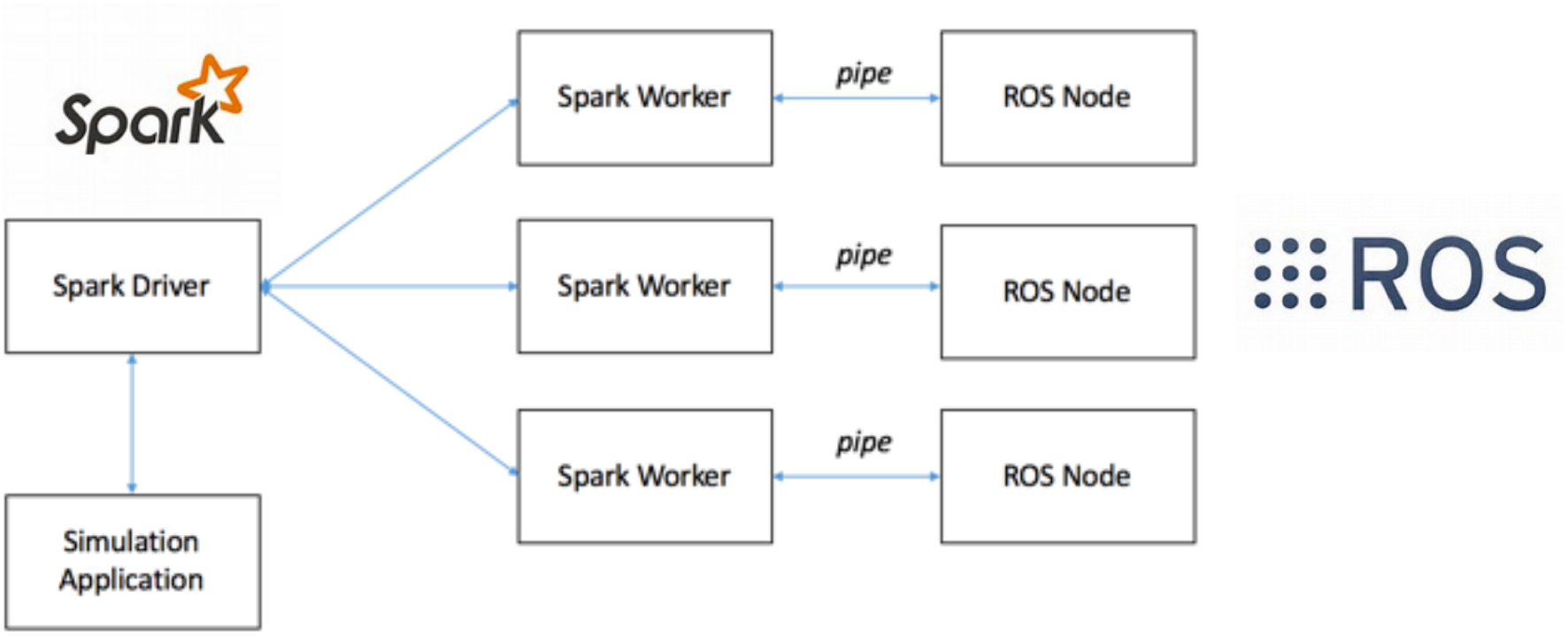
Cloud Infrastructure: Storage



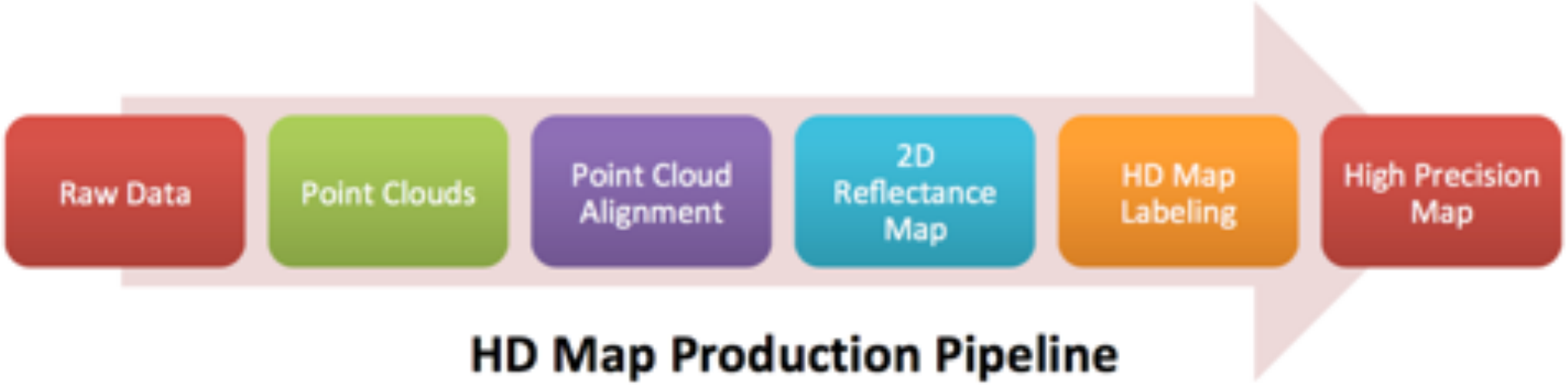


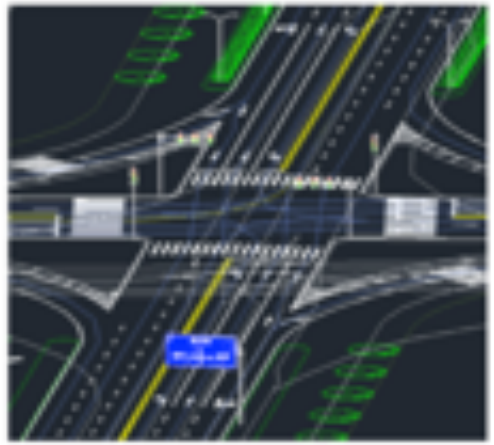
Cloud Infrastructure: Simulation



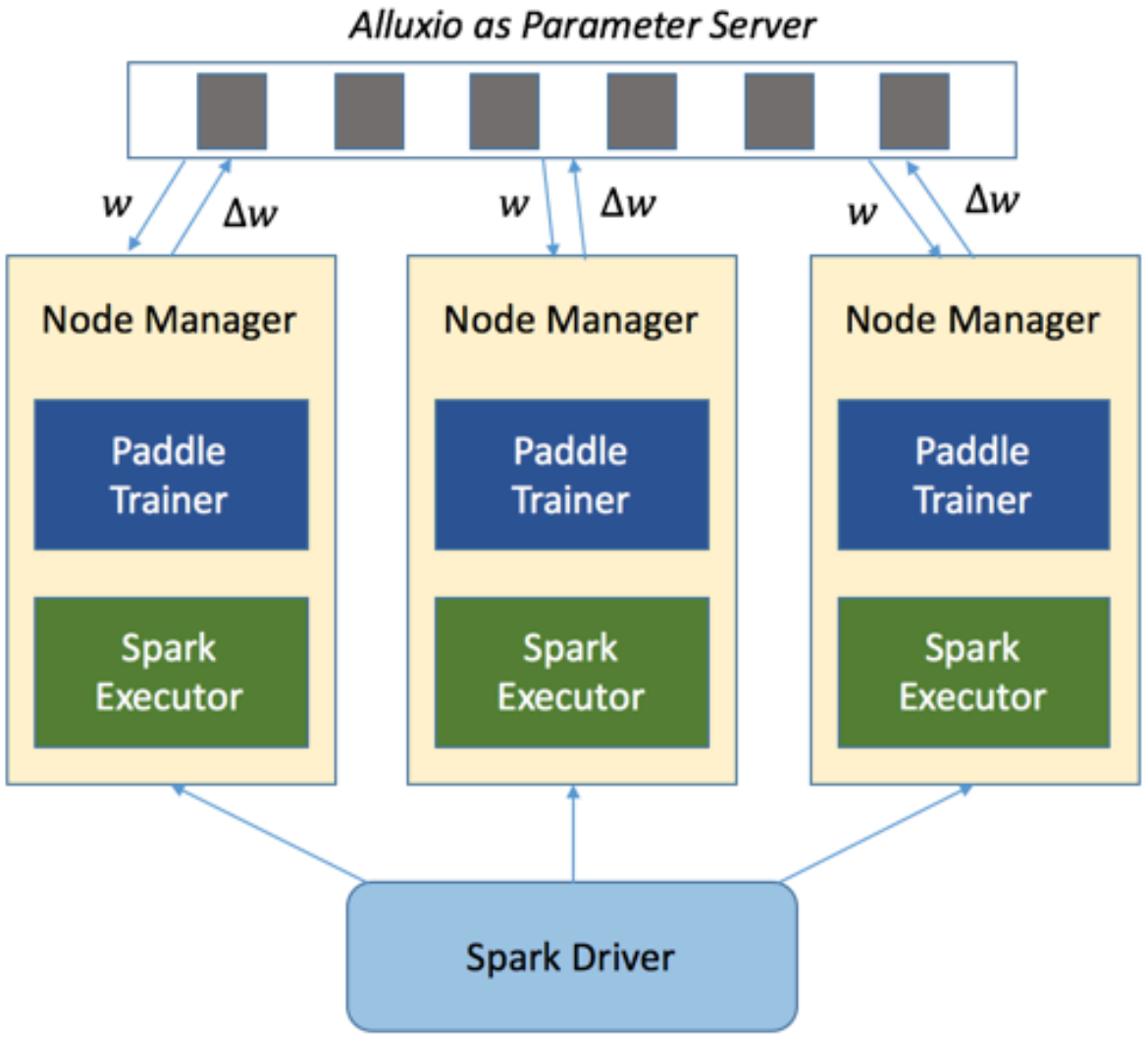


Cloud Infrastructure: HD Map

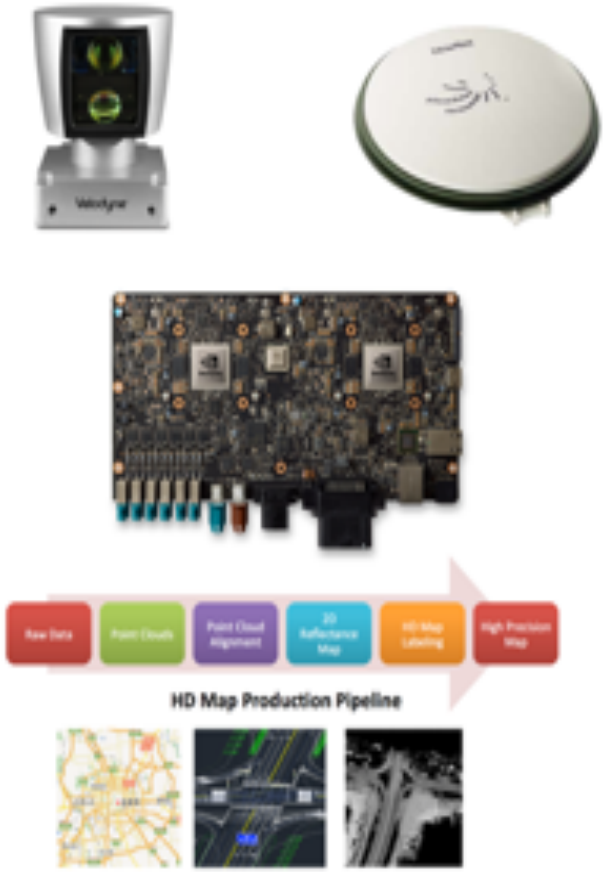




Cloud Infrastructure: Model Training



Problems with the Heavyweight Approach



> $100,000 USD Sensing Hardware Cost

> $30,000 USD Computing Hardware Cost

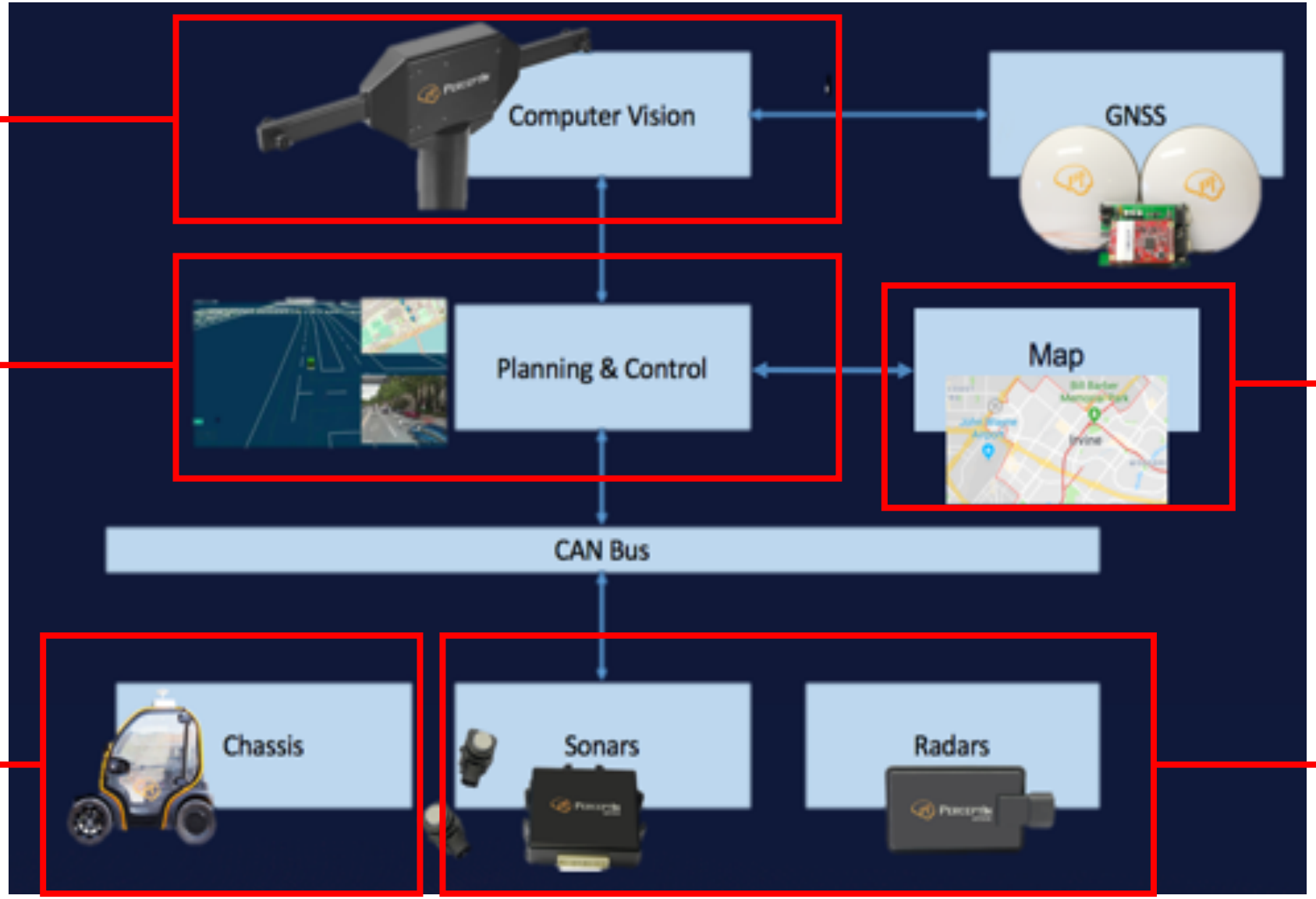
millions of USD to create a maintain a HD map

Outline

- Overview of Heavyweight Autonomous Driving
- Overview of Lightweight Autonomous Driving
- Edge Computing: Challenges and Opportunities
- Acceleration Example I: Fully Autonomous Robot Powered by a Cell Phone
- Acceleration Example II: FPGA-based ORB Feature Extraction for Real-Time Visual SLAM
- Acceleration Example III: Bundle Adjustment Acceleration on Embedded FPGAs with Co-observation Optimization
- Potential Class Projects

Computer Vision-Based Sensor Fusion

Modular Design Approach



Sensor Fusion for Active Perception and Localization

behavioral layer
MPC + PID

Open Street Map

throttle-by-wire
steer-by-wire
brake-by-wire

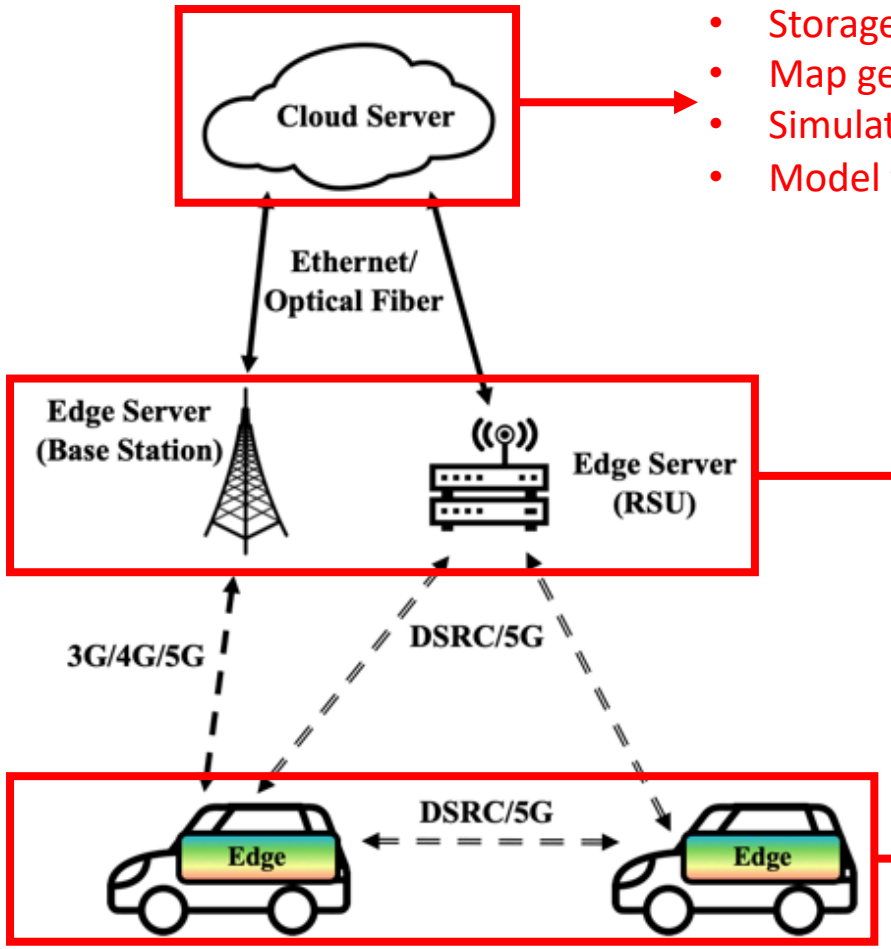
Passive Perception

Outline

- Overview of Heavyweight Autonomous Driving
- Overview of Lightweight Autonomous Driving
- Edge Computing: Challenges and Opportunities
- Acceleration Example I: Fully Autonomous Robot Powered by a Cell Phone
- Acceleration Example II: FPGA-based ORB Feature Extraction for Real-Time Visual SLAM
- Acceleration Example III: Bundle Adjustment Acceleration on Embedded FPGAs with Co-observation Optimization
- Potential Class Projects

Edge Computing for Autonomous Vehicles: Challenges

Liu, S., Liu, L., Tang, J., Yu, B., Wang, Y. and Shi, W., 2019. Edge Computing for Autonomous Driving: Opportunities and Challenges. Proceedings of the IEEE, 107(8), pp.1697-1716. Overview of Lightweight Autonomous Driving



- Storage and computing
- Map generation
- Simulation
- Model training

- Cooperative decisions
- Cooperative sensing

- Benchmarks
- Architectures
- Runtime
- RTOS
- Middleware

Security

Edge Client

Journals & Magazines > Proceedings of the IEEE > Volume: 107 Issue: 8

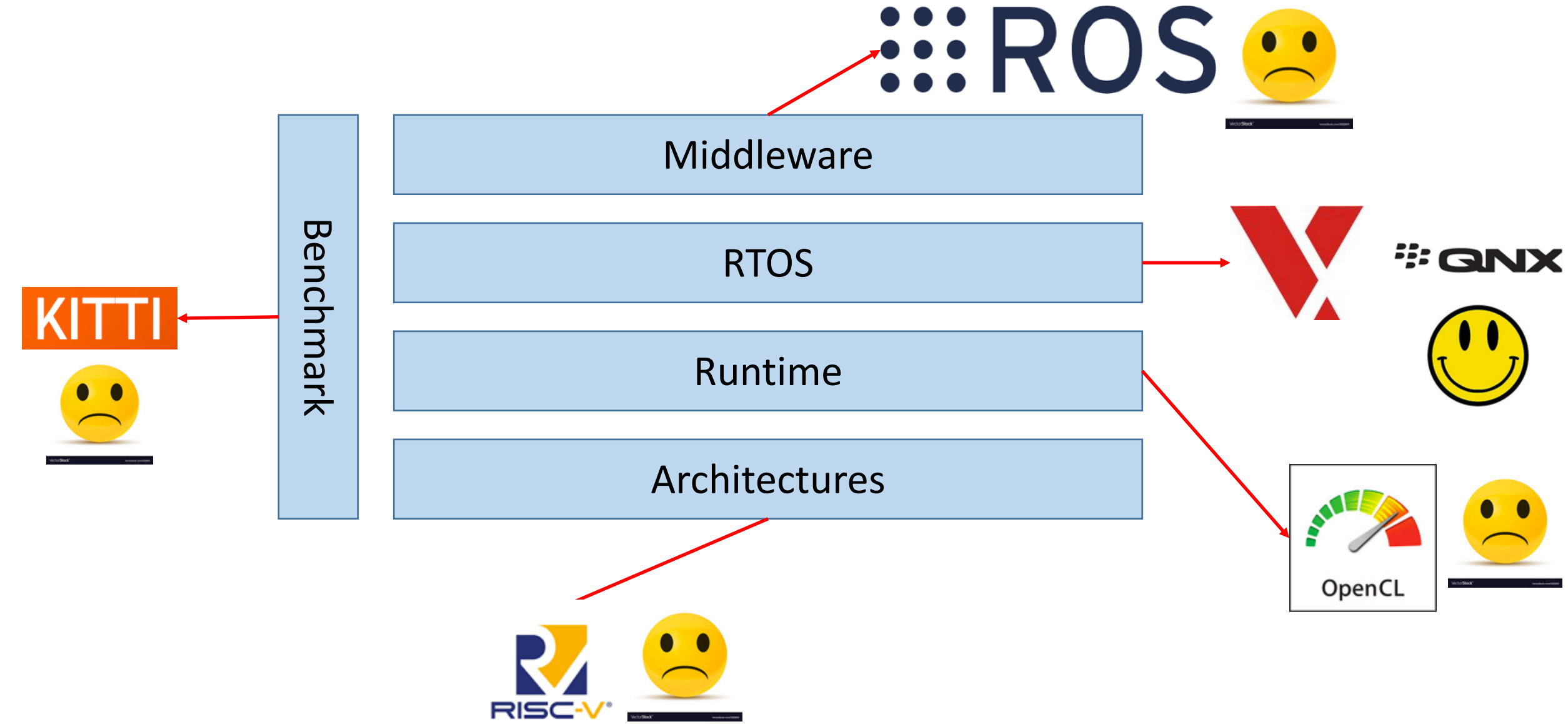
Edge Computing for Autonomous Driving: Opportunities and Challenges

Publisher: IEEE

6 Author(s) Shaoshan Liu ; Liangkai Liu ; Jie Tang ; Bo Yu ; Yifan Wang ; Weisong Shi View All Authors

Layer	Purpose	Proposed Solutions	Research Directions
Architecture	Hardware computing units to execute autonomous driving workloads	[1][37][38][39][40][41] [42][43]	accelerators for various autonomous driving workloads; cache and memory architecture design; non-volatile storage for critical data; hardware security
Runtime	Software layer to efficiently dispatch incoming tasks at run time to different computing units	[62][50][51][52][53][54] [55][56][58][59][60][61]	scheduler and dispatcher for highly heterogeneous computing systems; abstraction to hide low-level details; cloud awareness
Middleware	Software layer to enable complex interactions between autonomous driving services	[63][64][65][66][67][21] [68]	low overhead and memory footprint; edge-cloud interaction; security and reliability
Benchmark	Tools to evaluate edge computing systems	[36][26][27][28][29][30] [31][32][33][34][35]	more dynamic workloads and data to cover more usage scenarios; standardized, scoring methods to rank edge computing systems

Edge Client


ROS
Middleware
RTOS
Runtime
Architectures
Benchmark
KITTI
QNX
OpenCL
RISC-V

V2X

Research	Application Scenario	Proposed Solutions	Communication Protocol
IVHW	Safe driving	Warning message are transmitted as broadcast message, and vehicle takes a local decision-making strategy.	Frequency band of 869MHz
FleetNet	Safe driving, internet protocol based applications	Uses ad-hoc networking to support multi-hop inter-vehicle communications, proposes a position-based forwarding mechanism	IEEE 802.11 Wireless LAN
CarTALK 2000	Cooperative driver assistance applications	Uses ad-hoc communication network to support co-operative driver assistance applications, a spatial aware routing algorithm which takes some spatial information like underlying road topology into consideration to solve	IEEE 802.11 Wireless LAN
AKTIV	Safe driving	WLAN technology is that the latency of safety related applications required to be less than 500ms	Cellular systems
WILLWARN	Warning applications	Propose a risk detection approach based on in-vehicle data. The warning message includes obstacles, road conditions, low visibility, and construction sites. A decentralized distribution algorithm to transmit the warning message to vehicles approaching the danger spot through V2V communication	IEEE 802.11 Wireless LAN
NoW	Mobility and internet applications	A hybrid forwarding scheme considering both network-layer and application-layer is developed. Also, some security and scalability issues are discussed.	IEEE 802.11 Wireless LAN
SAFESPOT	Safe driving	SAFESPOT is an integrated project which aims at using roadside infrastructure to improve driving safety. detects dangerous situations and shares the warning messages in real time	IEEE 802.11 Wireless LAN
sim^{TD}	Traffic Manipulation, safe driving, and internet based applications	Real environment deployment of the whole Intelligent Transportation System. The system architecture of simTD can be divided into three parts: ITS vehicle station, ITS roadside station, and ITS central station.	IEEE 802.11p (DSRC)

Security

Journals & Magazines > IEEE Transactions on Intellig... > Volume: 16 Issue: 2

Potential Cyberattacks on Automated Vehicles

Publisher: IEEE

2 Author(s) Jonathan Petit ; Steven E. Shladover View All Authors

Security Category	Security Threats	Defense Technologies
Sensors	Spoofing cameras by fake traffic objects.	**Multi-sensor data fusion**: System check and correct the sensor data from multiple sources.
	Jamming GPS receiver by high-power false GPS transmitter.	
	Jamming IMU sensor by powerful magnetic field.	
	Jamming LiDAR by light laser pulse.	
	Jamming and Spoofing ultrasonic sensors and MMW radars by specific signal generator.	
Operating System	Hijacking ROS node to consume system resources.	**Linux container**: Use the container technology to throttle the resource utilization of each ROS node. **Trusted execution environment**: Run the key ROS node in a trusted execution environment.
	Hijacking ROS node to send manipulated messages.	
	Sniffing ROS message to steal private data.	
	Repeating the intercepted ROS message to disturb other ROS nodes.	
Control System	Hijacking CAN bus by OBD-II port.	**Message encryption**: Encrypt message in CAN bus.
	Hijacking CAN bus by media player.	
	Hijacking CAN bus by Bluetooth.	
	Injecting manipulated messages on CAN bus.	
	DoS attack on CAN bus.	
V2X	DoS and DDoS attack on vehicle and infrastructure.	**Authentication and certification**: The node access the V2X network should be authenticated and provide security certificates and keys.
	Sybil attack by creating multiple fake vehicles in road.	
	Sniffing private data by short-range wireless protocol.	
	Broadcasting fake traffic information to nearby vehicles.	

Outline

- Overview of Heavyweight Autonomous Driving
- Overview of Lightweight Autonomous Driving
- Edge Computing: Challenges and Opportunities
- Acceleration Example I: Fully Autonomous Robot Powered by a Cell Phone
- Acceleration Example II: FPGA-based ORB Feature Extraction for Real-Time Visual SLAM
- Acceleration Example III: Bundle Adjustment Acceleration on Embedded FPGAs with Co-observation Optimization
- Potential Class Projects

Samsung Galaxy S7 Powered by Snapdragon 820

CPU

Up to 2.2 GHz quad-core (Quad-core custom 64-bit Qualcomm® Kryo™)

GPU

Qualcomm® Adreno™ 530 GPU

Up to OpenGL ES 3.1+

DSP

Qualcomm® Hexagon™ 680 DSP

android NDK

OpenCL

Hexagon DSP SDK

Project in Early 2016

Profiling Results

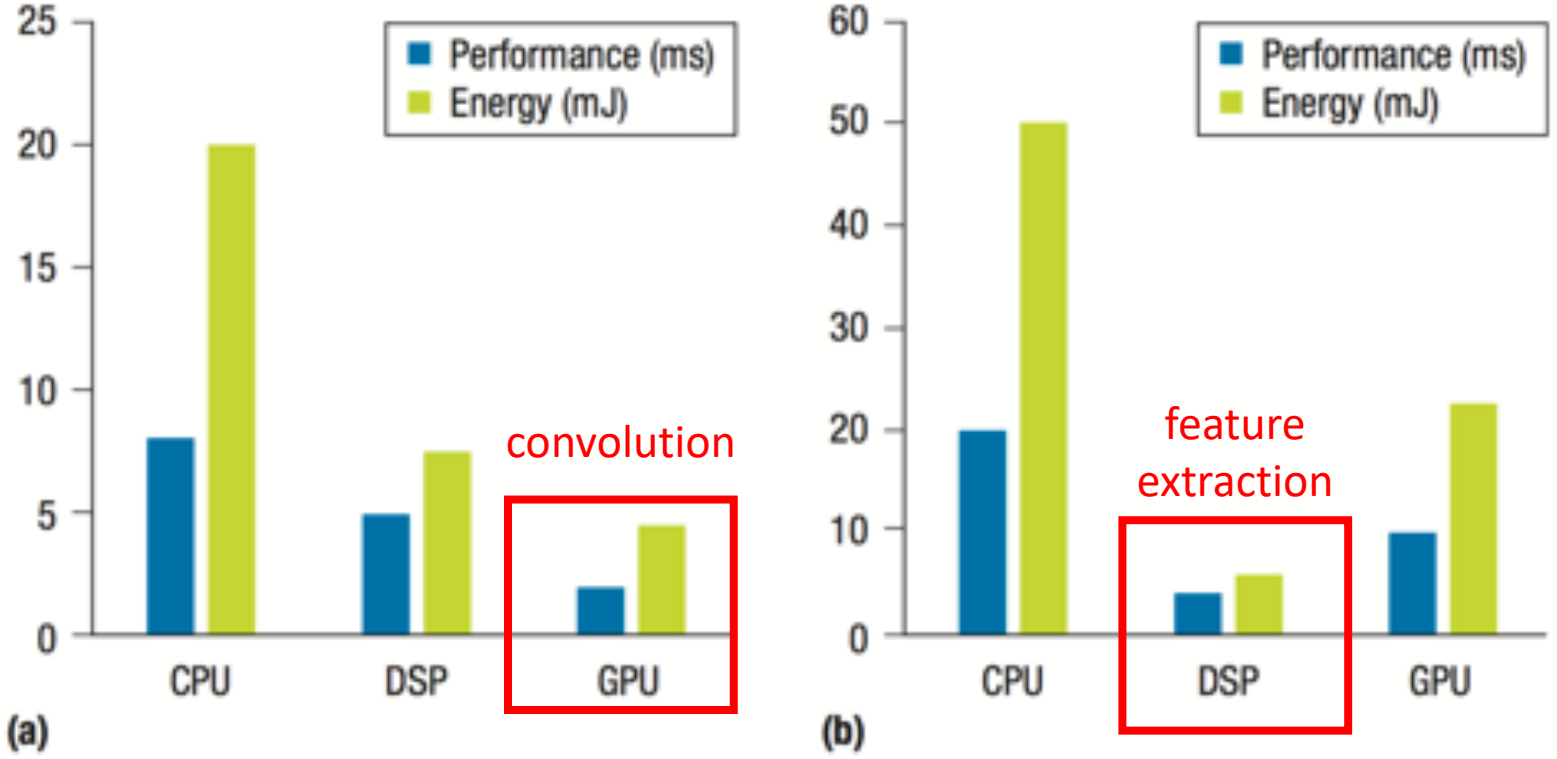
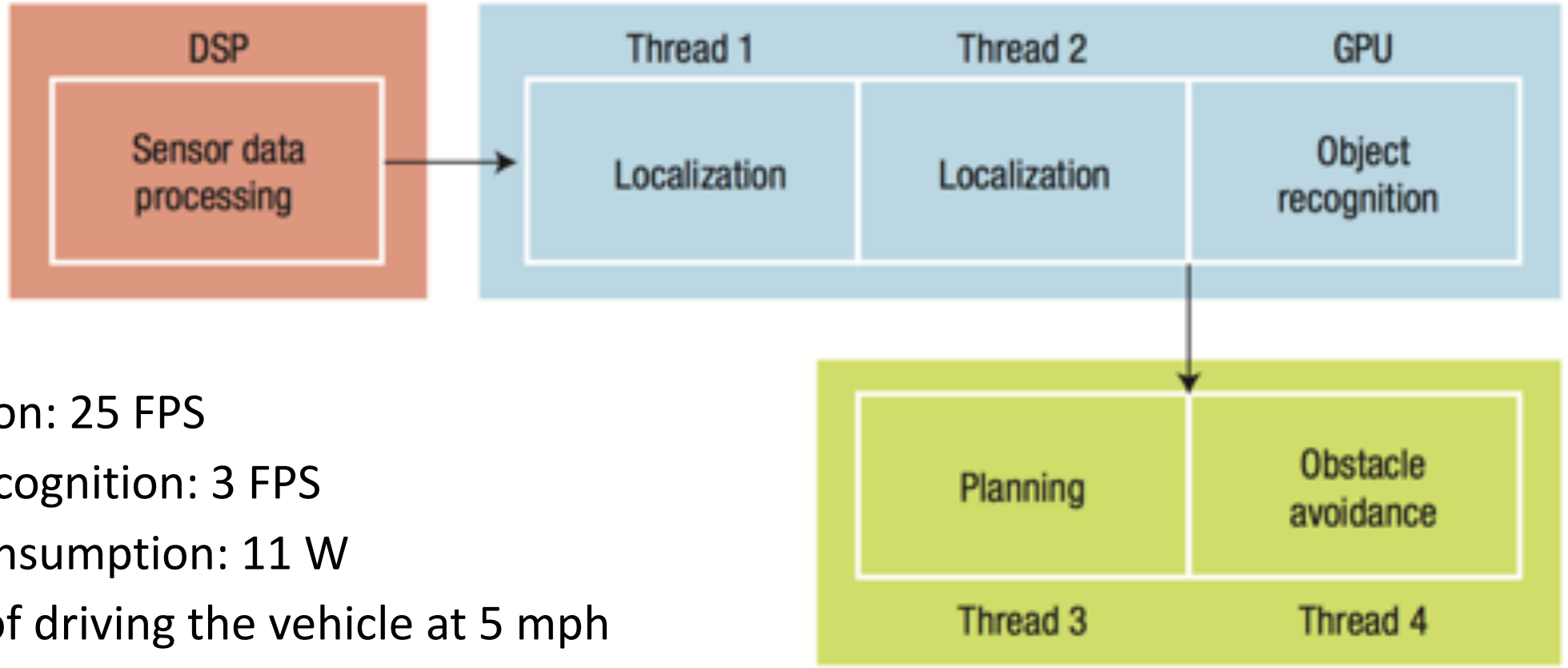


FIGURE 2. Performance and energy in (a) convolution and (b) feature-extraction tasks. In (a), the GPU takes only 2 ms and uses only 4.5 millijoules (mJ) to complete convolution tasks. In (b), the digital signal processor (DSP) is the most efficient unit for feature extraction, taking 4 ms and consuming only 6 mJ to complete a task.

Mobile SoC:

- Quad-core CPU @ 2.2 GHz
- Hexagon 680 DSP
- Adreno 530 GPU
- Peak power ~ 15 W

Heterogeneous Computing



- Localization: 25 FPS
- Object recognition: 3 FPS
- Power consumption: 11 W
- Capable of driving the vehicle at 5 mph

PERCEPTIN

普思英察

Outline

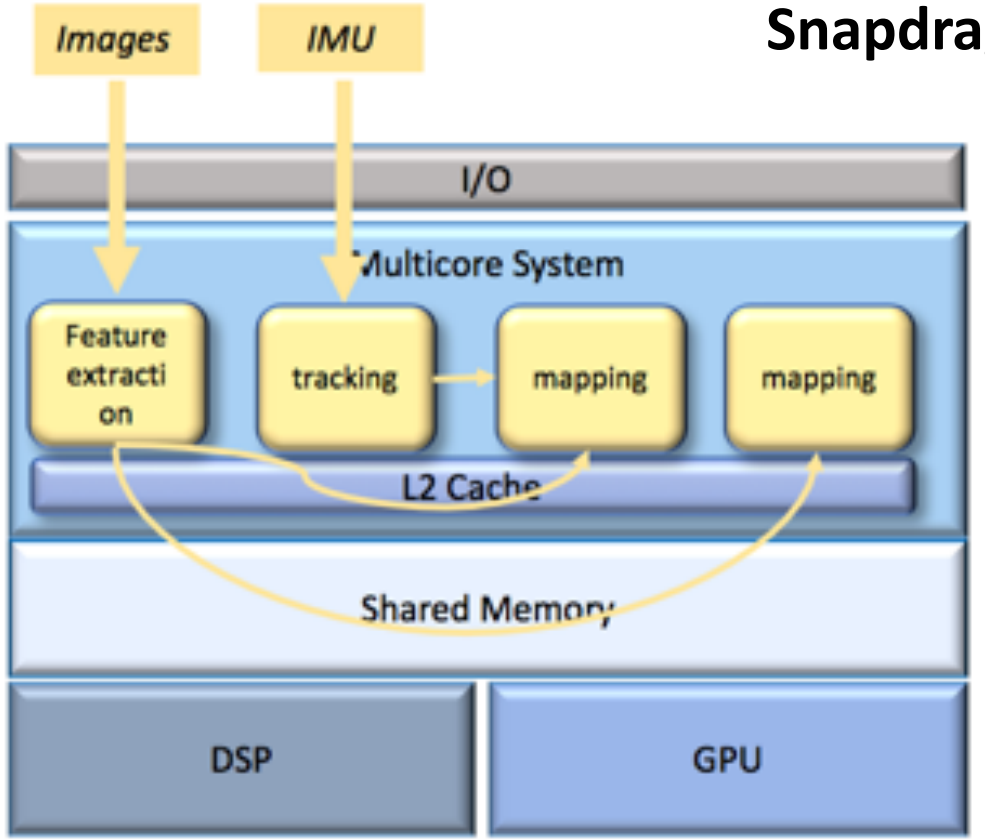
- Overview of Heavyweight Autonomous Driving
- Overview of Lightweight Autonomous Driving
- Edge Computing: Challenges and Opportunities
- Acceleration Example I: Fully Autonomous Robot Powered by a Cell Phone
- Acceleration Example II: FPGA-based ORB Feature Extraction for Real-Time Visual SLAM
  - *Building Block: Feature Extraction*
- Acceleration Example III: Bundle Adjustment Acceleration on Embedded FPGAs with Co-observation Optimization
- Potential Class Projects

VSLAM on Multi-Core Embedded SoC

Tang, J., Yu, B., Liu, S., Zhang, Z., Fang, W. and Zhang, Y., 2018, October. π-SoC: Heterogeneous SoC Architecture for Visual Inertial SLAM Applications. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (pp. 8302-8307). IEEE.

20 FPS @ 8 W with parallel computing

Snapdragon 820



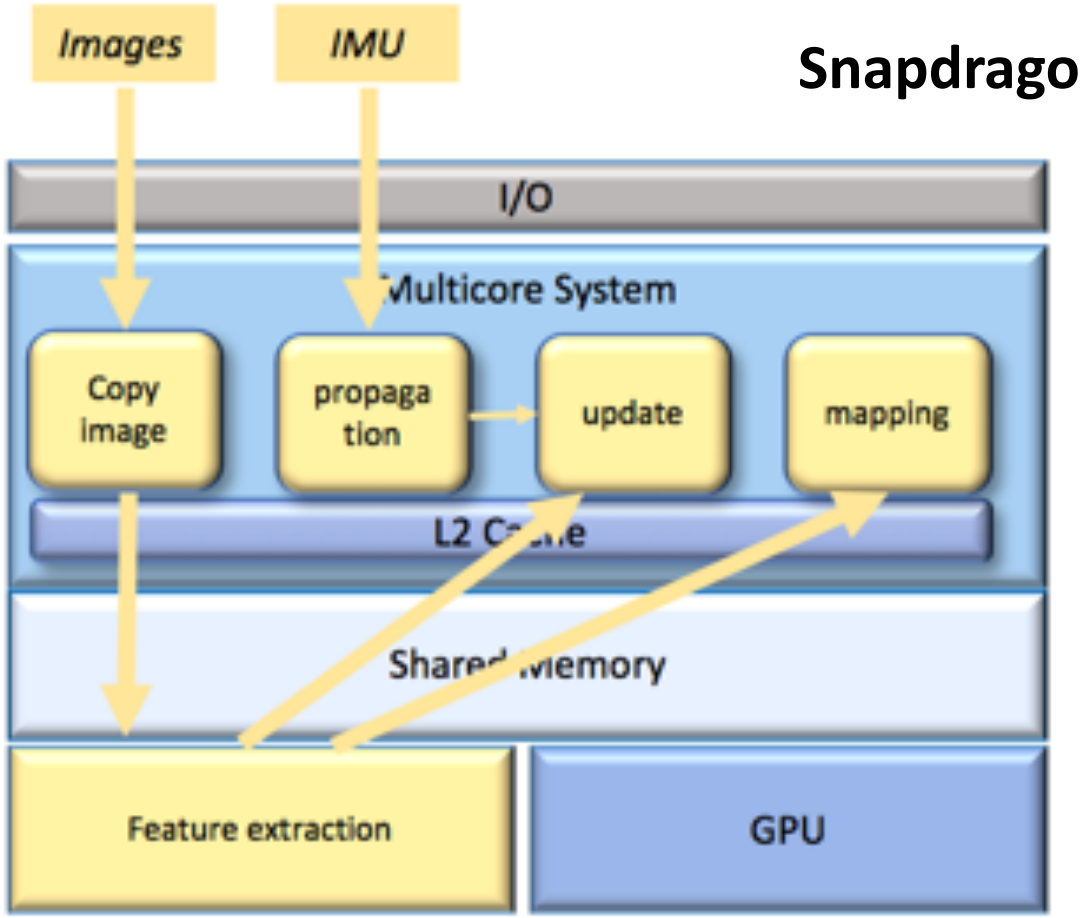
Performance and Power

Feature Extraction:	45 ms
Propagation:	2 ms
Update:	30 ms
Mapping:	15 ms
Average Power:	8 W

PERCEPTIN

VSLAM on Multi-Core Embedded SoC

30 FPS @ 9 W with parallel and heterogeneous computing



Snapdragon 820

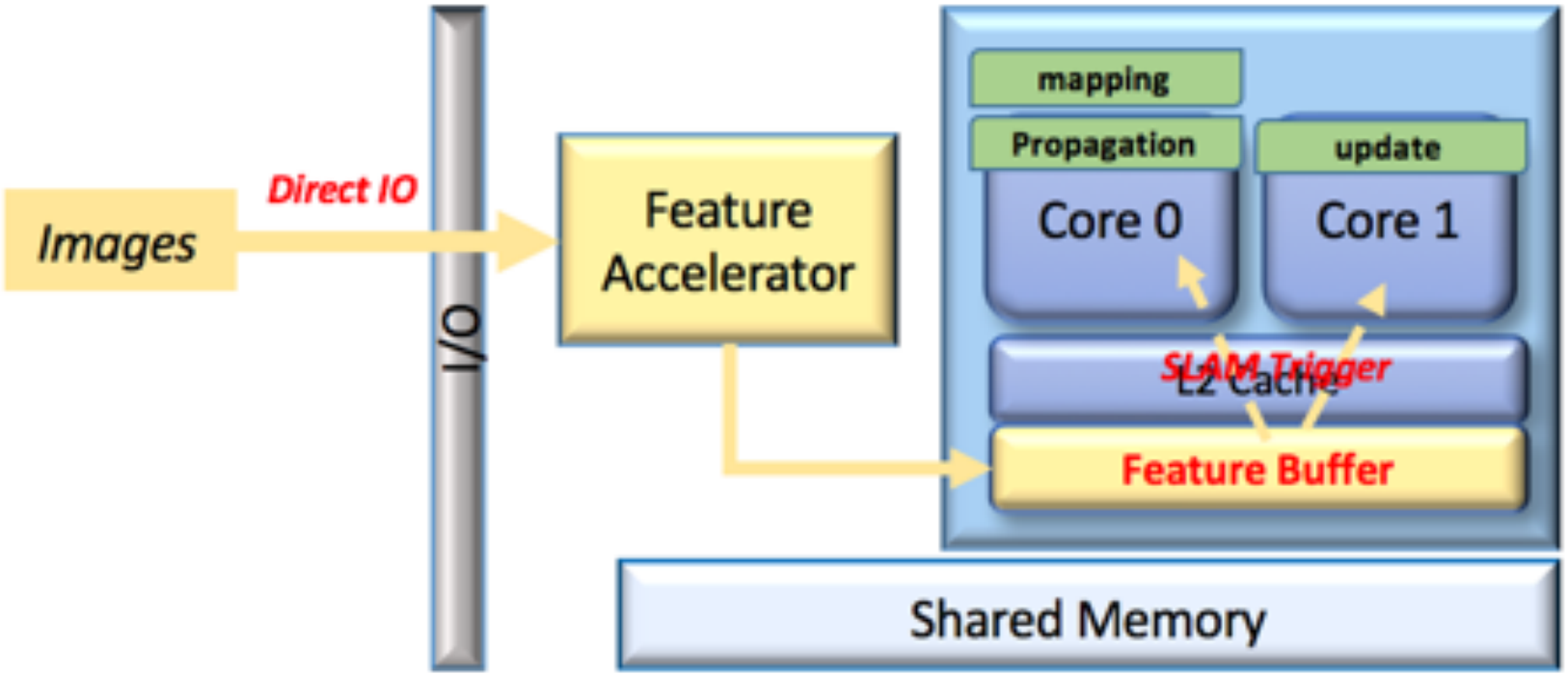
Performance and Power

Feature Extraction:	20 ms
Propagation:	2 ms
Update:	30 ms
Mapping:	15 ms
Average Power:	9 W

PERCEPTIN

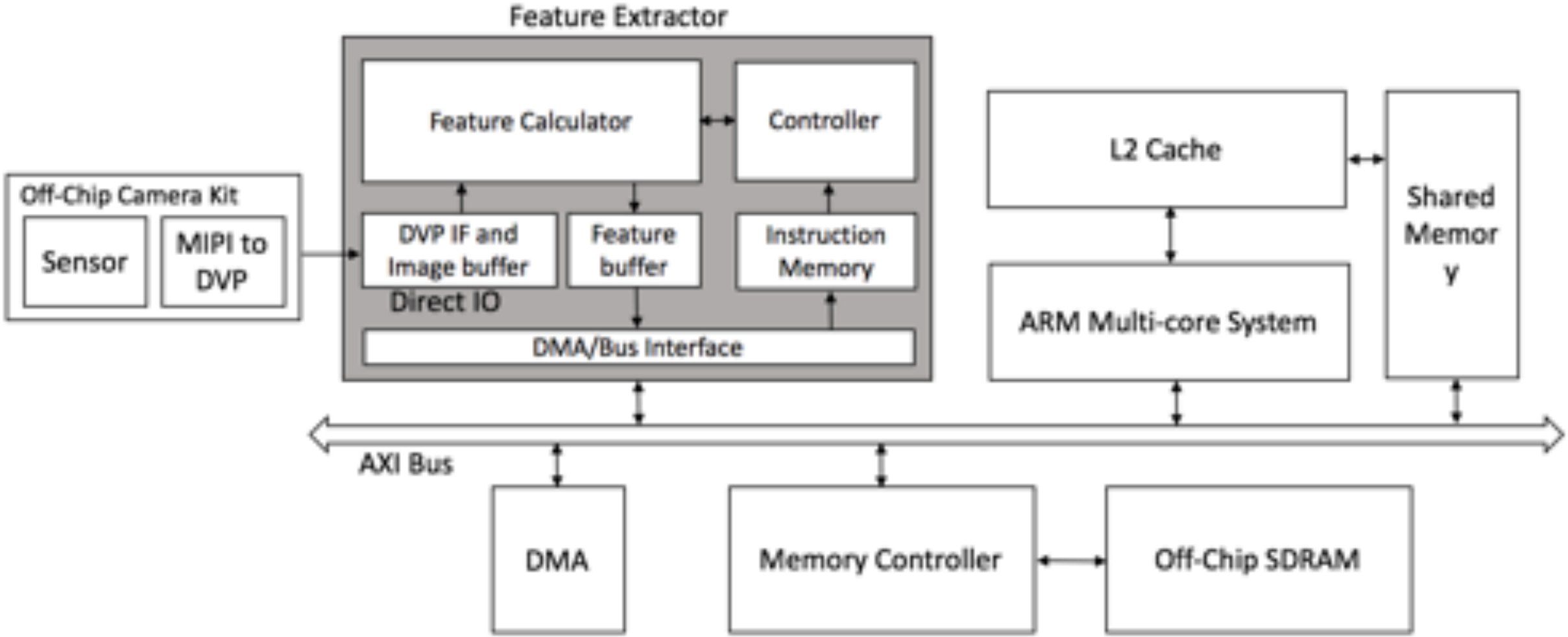
π-SoC Architecture

Problems: Computing, IO, Memory



PERCEPTIN

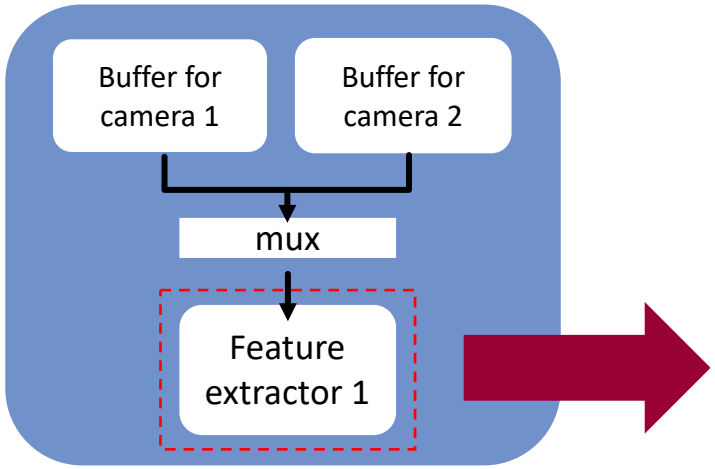
FPGA Implementations of π-SoC Architecture

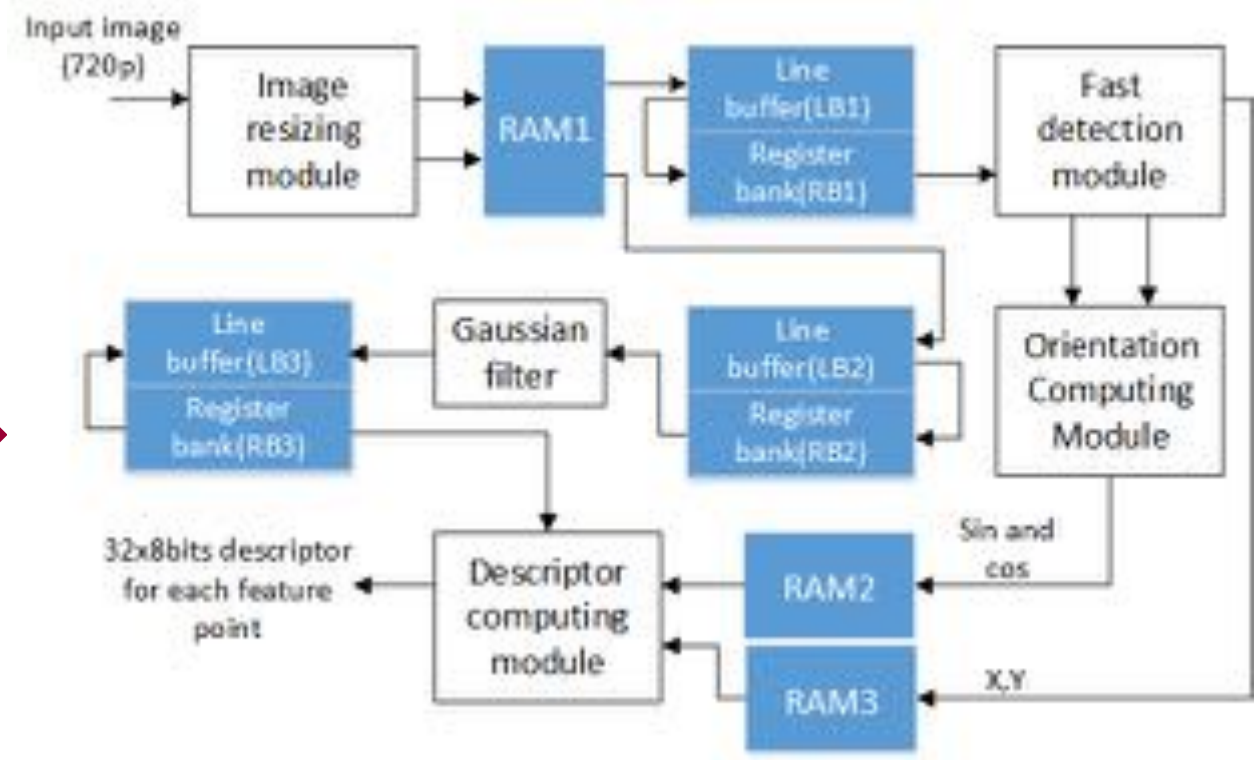


PERCEPTIN

Image Processing Frontend Acceleration

- Architecture for feature extractor

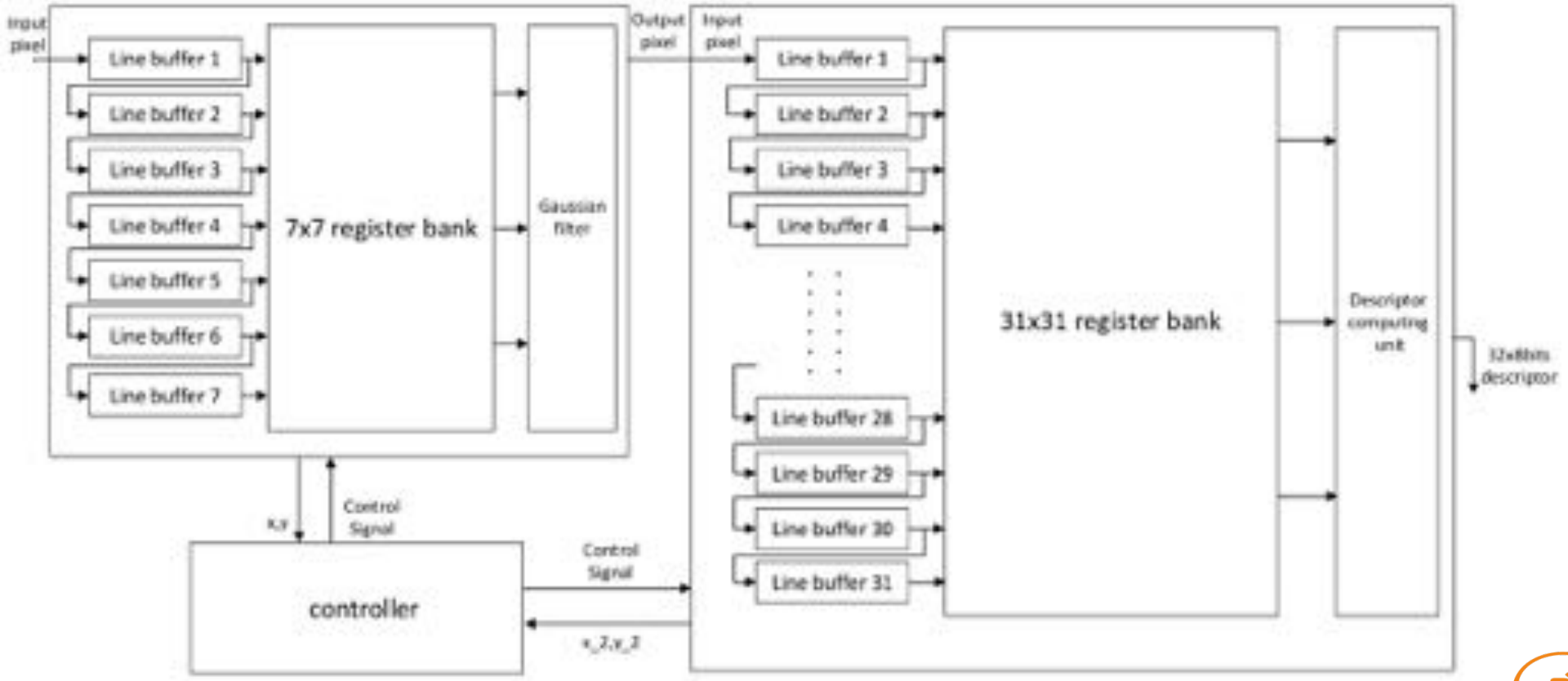




PERCEPTIN

Image Processing Frontend Acceleration

- Synchronized two-stages shifting line buffers
  - Compute Gaussian filtering and descriptors in a data driven way
  - Control unit synchronizes the result of Gaussian filter and feaure coordinate and orientation



PERCEPTIN

Evaluation Results

- VSLAM front end system
  - 2 feature extractors (frame multiplexing), 2 feature matchers and external control logic and memory
- Compared with software solution

	Power (W)	Performance (FPS)	Power (%)	Performance (%)
π-SoC	2.31	42	--	--
Nvidia TX1	7	9	3x	21%
Intel Core i7	80	15	34x	36%

- **3x more power efficient and delivers 5x of computing power compared to Nvidia TX1**
- **34x more power efficient and delivers 3x of computing power compared Intel Core i7**

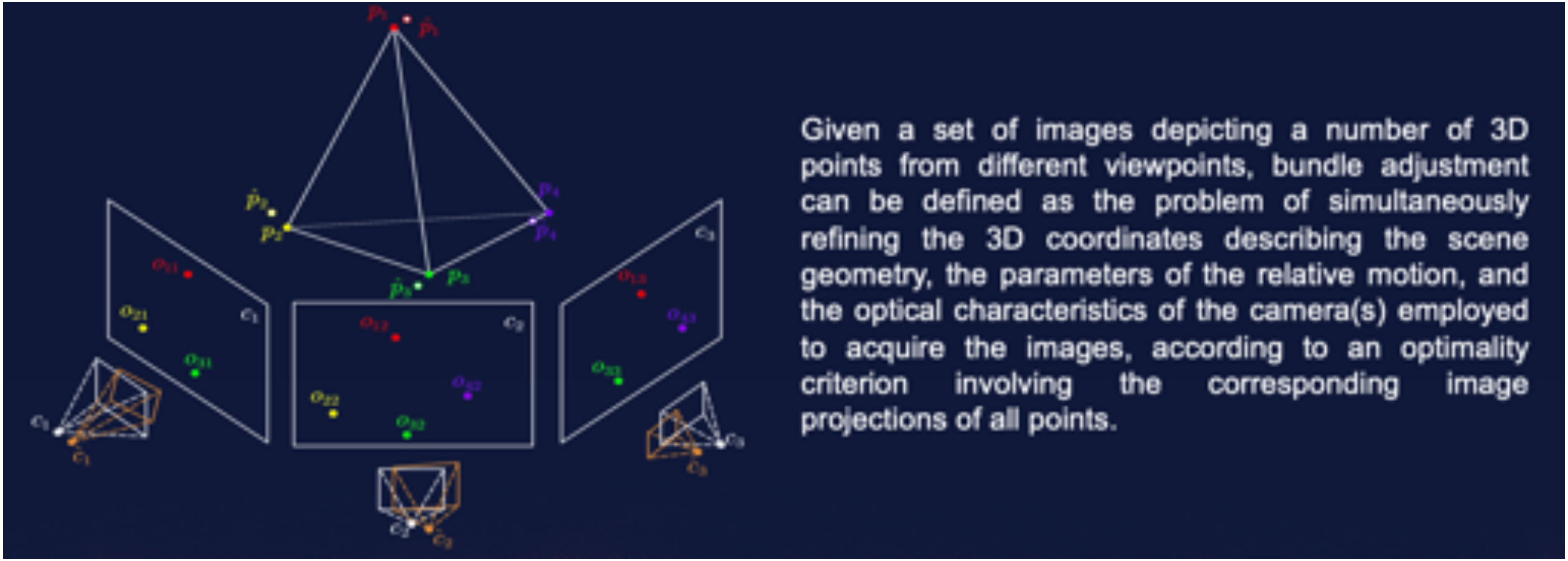
PERCEPTIN

Outline

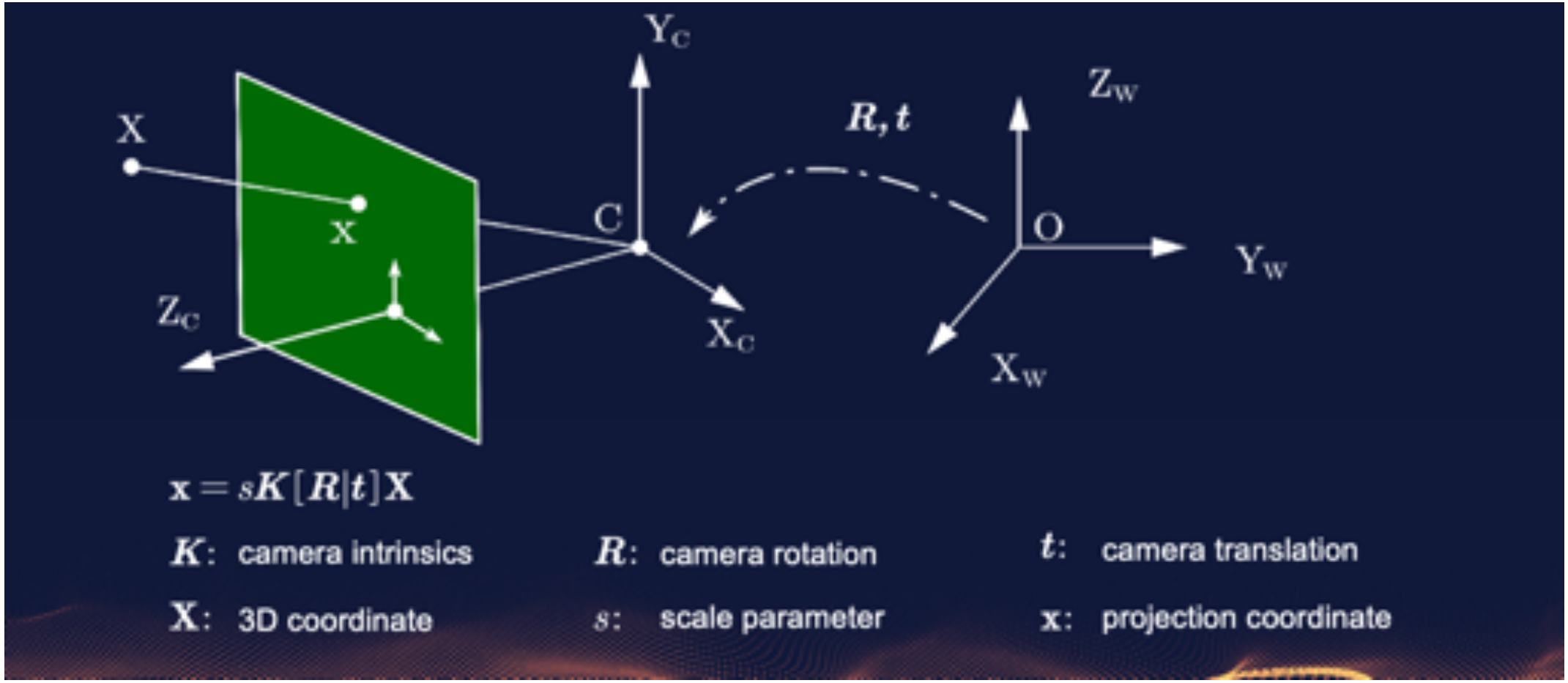
- Overview of Heavyweight Autonomous Driving
- Overview of Lightweight Autonomous Driving
- Edge Computing: Challenges and Opportunities
- Acceleration Example I: Fully Autonomous Robot Powered by a Cell Phone
- Acceleration Example II: FPGA-based ORB Feature Extraction for Real-Time Visual SLAM
- Acceleration Example III: Bundle Adjustment Acceleration on Embedded FPGAs with Co-observation Optimization
  - *Building Block: Bundle Adjustment*
- Potential Class Projects

Bundle Adjustment

Qin, S., Liu, Q., Yu, B. and Liu, S., 2019, April. π-BA: Bundle Adjustment Acceleration on Embedded FPGAs with Co-observation Optimization. In 2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM) (pp. 100-108). IEEE.



Camera Projection：3D points to 2D image points

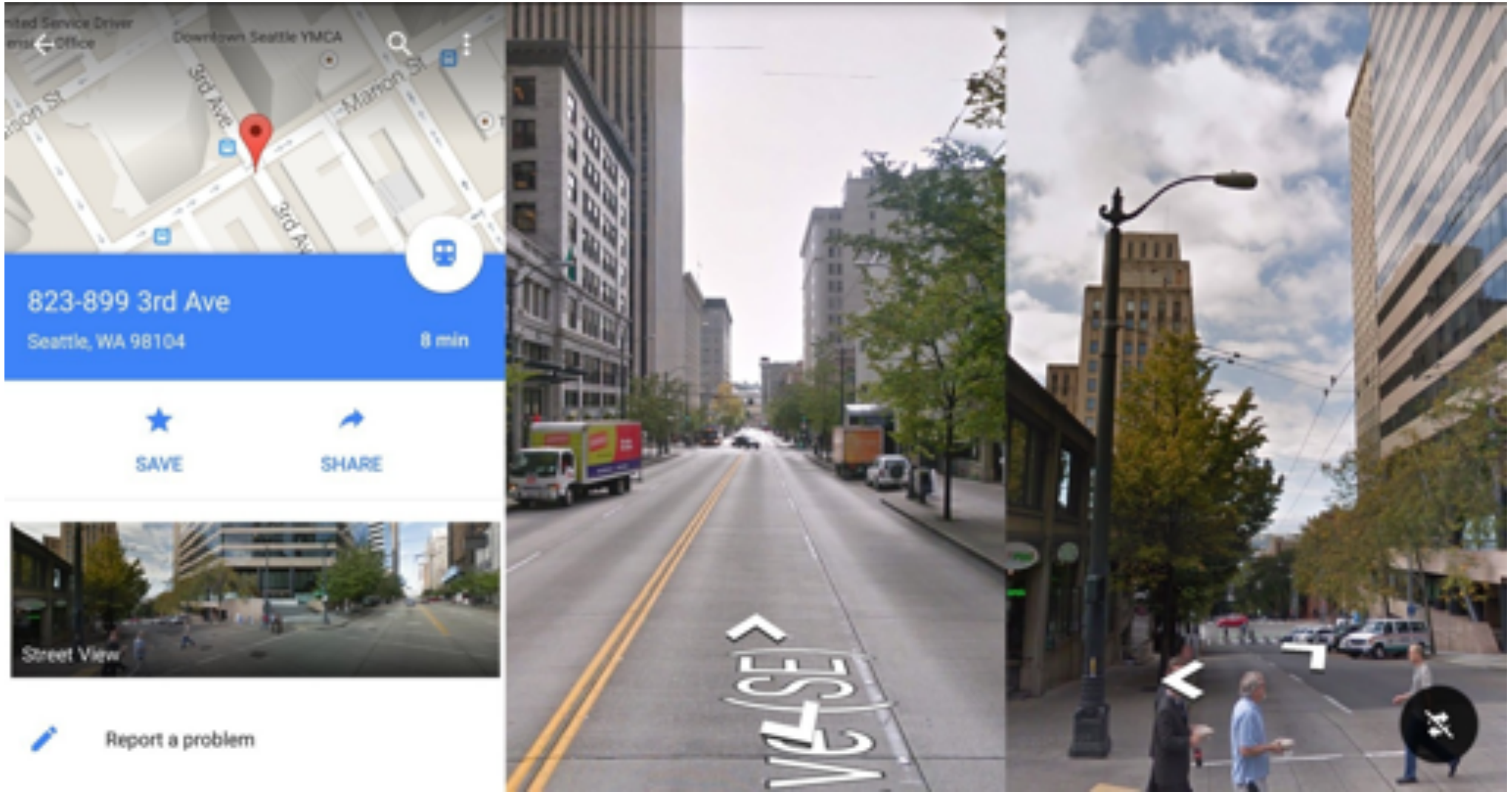


BA Application: 3D Scene Reconstruction

S.Agarwal,N.Snavely,I.Simon,S.M.Seitz, and R.Szeliski, “Building Rome in a day,” in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 72–79.

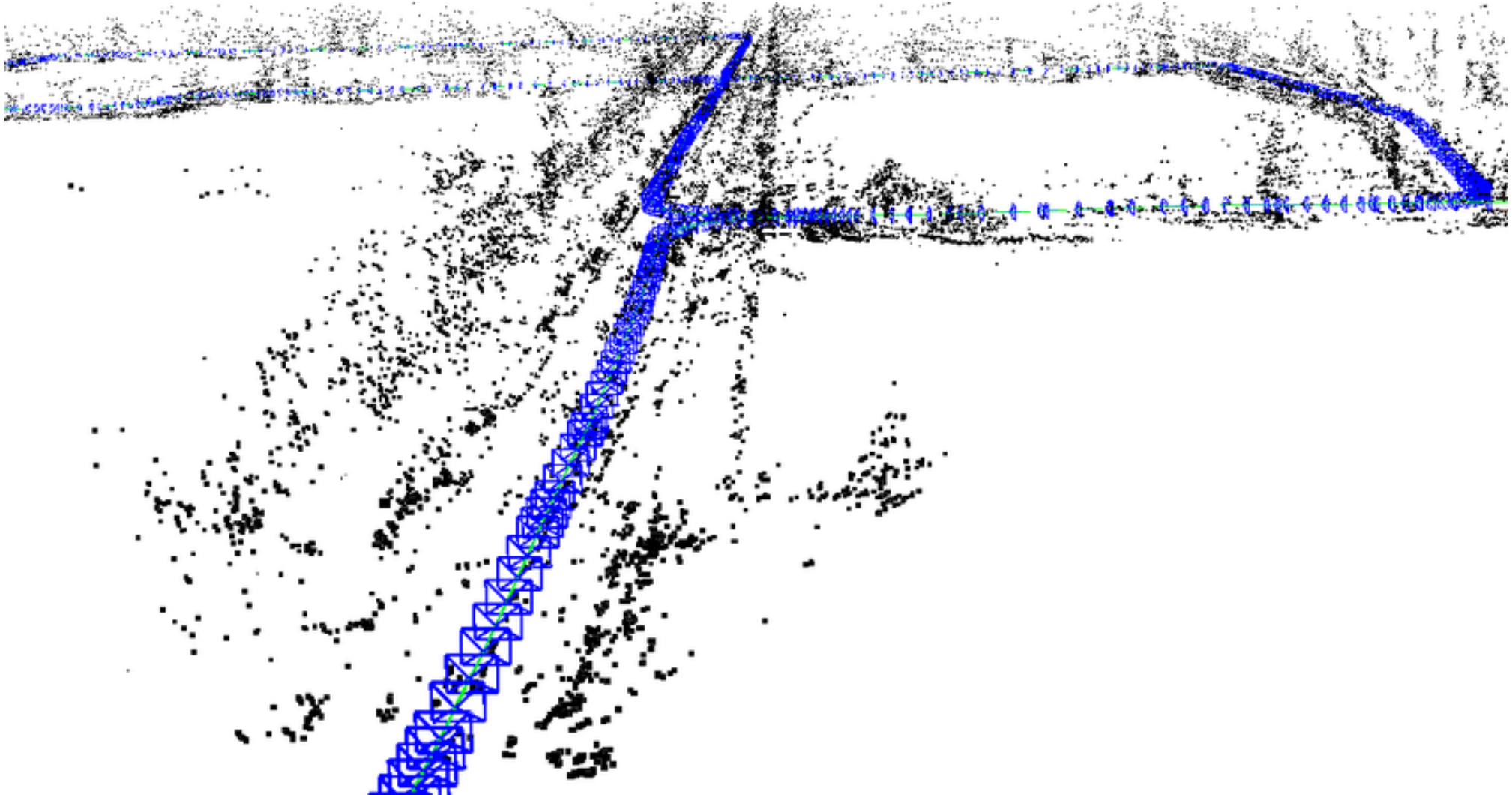
BA Application: Street View Generation

B. Klingner, D. Martin, and J. Roseborough, “Street view motion-from-structure-from-motion,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 953–960.



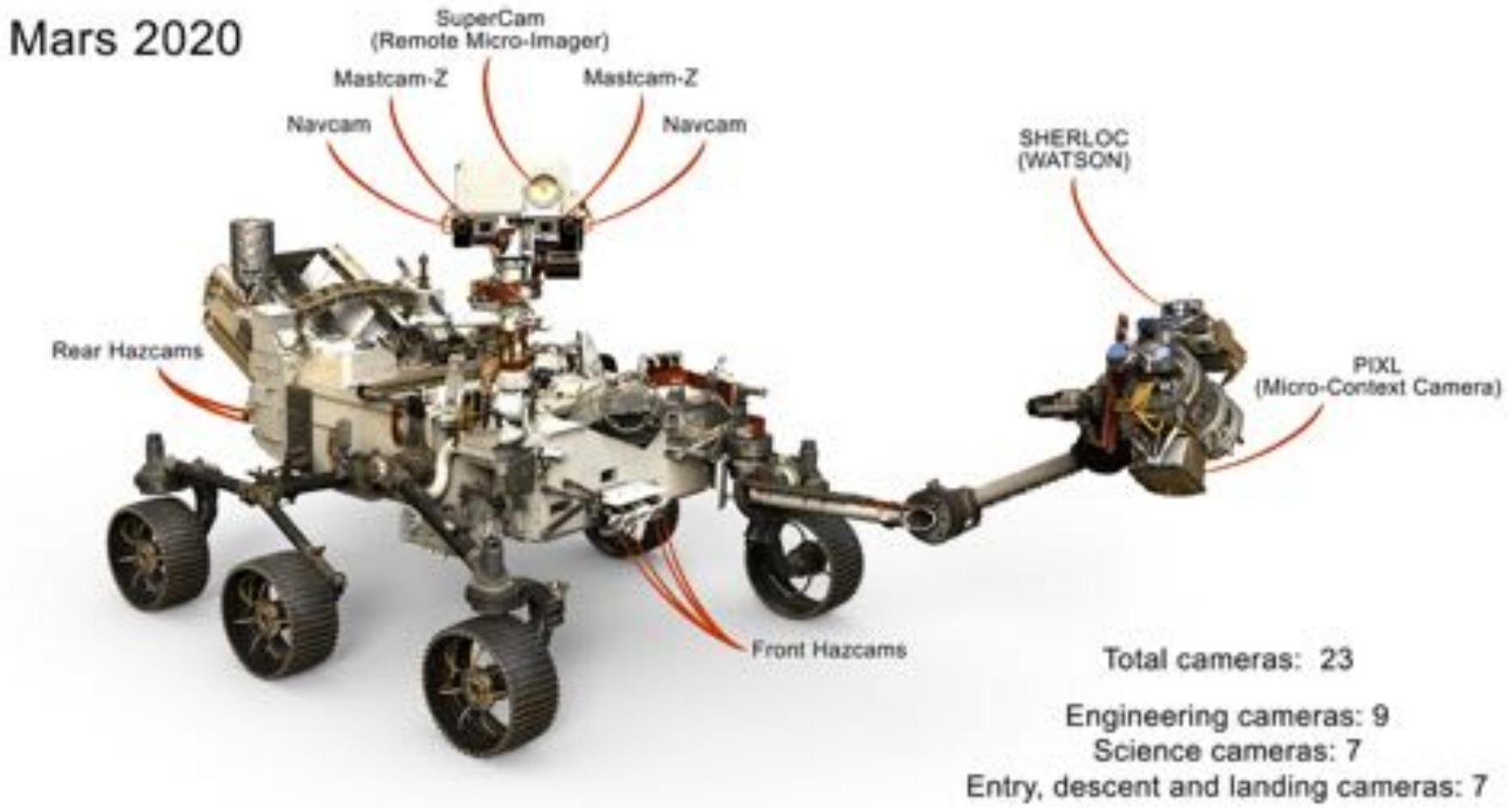
BA Application: Robot Localization

R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, “ORB-SLAM: a versatile and accurate monocular SLAM system,” *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.



BA Application: Space Exploration

M.Maimone,Y.Cheng,andL.Matthies,"Two years of visual odometry on the Mars exploration rovers," *Journal of Field Robotics*, vol. 24, no. 3, pp. 169–186, 2007.



BA Application: Space Exploration

Due to limiting processing power, Mars rover has to send images back to Earth for Bundle Adjustment processing!

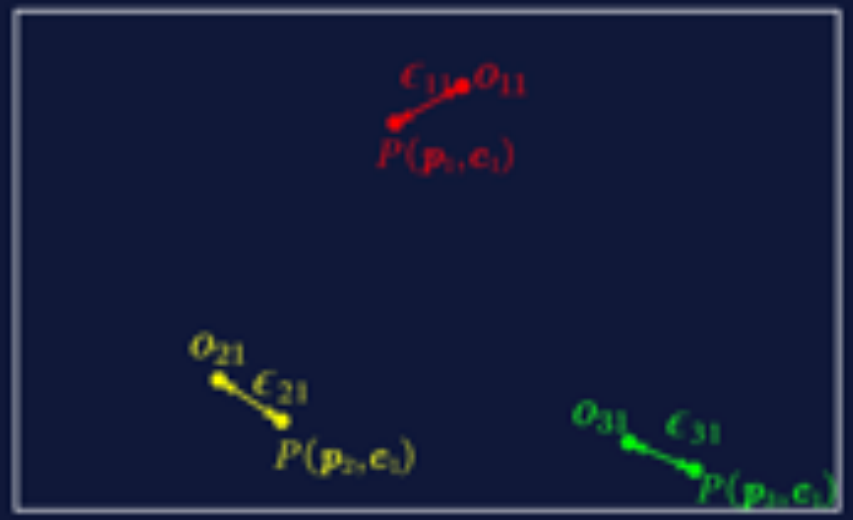
1. Rover captures images on Mars surface

2. Images sent to Earth for processing

3. Bundle Adjustment on Earth

4. Processed map sent back to Mars to aid rover navigation

Levenberg-Marquardt Method for Bundle Adjustment



Projection Function $\boldsymbol{x} = s\boldsymbol{K}[\boldsymbol{R}|\boldsymbol{t}]\boldsymbol{X} = P(\hat{\boldsymbol{p}},\hat{\boldsymbol{c}})$

Projection Residual $\boldsymbol{\epsilon}_{ij} = \boldsymbol{o}_{ij} - P(\boldsymbol{p}_i,\boldsymbol{c}_j)$

Cost Function

$$\min_{\boldsymbol{p}_i,\boldsymbol{c}_j} \sum_{i=1}^{a}\sum_{j=1}^{b} \sigma_{ij} \| \boldsymbol{o}_{ij} - P(\boldsymbol{p}_i,\boldsymbol{c}_j) \|$$

Inputs:

A vector function $f: \mathbb{R}^m \to \mathbb{R}^n$ with $n > m$

A measurement vector $\mathbf{x} \in \mathbb{R}^n$

Initial parameter $\mathbf{p_0} \in \mathbb{R}^m$

Outputs:

$\mathbf{p}^+ \in \mathbb{R}^m$ minimizing $\|\mathbf{x} - f(\mathbf{p})\|^2$

Where

f: $P(\boldsymbol{p}_i,\boldsymbol{c}_j)$

$$\mathbf{x} = (o_{11}, o_{12}, \cdots, o_{1b}, o_{21}, o_{22}, \cdots, o_{2b}, \cdots, o_{a1}, o_{a2}, , o_{\cdots ab})$$

$$\mathbf{p} = (\mathbf{p}_1^p, \mathbf{p}_2^p, \cdots \mathbf{p}_a^p, \mathbf{p}_1^c, \mathbf{p}_2^c, \cdots \mathbf{p}_b^c) = (\mathbf{p}^p, \mathbf{p}^c)$$

Levenberg-Marquardt Method in Detail

1. $k:=0;v:=2;\mathbf{p}:=\mathbf{p}_0;$
2. $\mathbf{J}:=f'(\mathbf{p});\epsilon:=\mathbf{x}-f(\mathbf{p});$

Jacobian Matrix Initial

3. $\mathbf{A}:=\mathbf{J}^{\mathrm{T}}\mathbf{J};\mathbf{D}^{\mathrm{T}}\mathbf{D}=\mathrm{diag}(\mathbf{A});\mathbf{g}:=\mathbf{J}^{\mathrm{T}}\epsilon;$
4. $\mathrm{stop}:=(\|\mathbf{g}\|_\infty\le\varepsilon_1);\mu:=\tau^*\max_{i=1,\cdots,m}(A_{ii})$
5. while (not stop) and $(k<k_{\max})$
6. $k:=k+1;$
7. $\mathrm{Solve}(\mathbf{A}+\mu\mathbf{D}^{\mathrm{T}}\mathbf{D})\delta_{\mathbf{p}}=\mathbf{g};$

Solving Linear Function

Schur Elimination

$$\mathbf{S}=\mathbf{J}^{c\mathrm{T}}\mathbf{J}^{c}-\mathbf{J}^{c\mathrm{T}}\mathbf{J}^{p}(\mathbf{J}^{p\mathrm{T}}\mathbf{J}^{p})^{-1}\mathbf{J}^{p\mathrm{T}}\mathbf{J}^{c};$$

$$\mathbf{r}=\mathbf{J}^{c\mathrm{T}}\epsilon-\mathbf{J}^{c\mathrm{T}}\mathbf{J}^{p}(\mathbf{J}^{p\mathrm{T}}\mathbf{J}^{p})^{-1}\mathbf{J}^{p\mathrm{T}}\epsilon;$$

$$\mathrm{Solve}(\mathbf{S}\delta\mathbf{p}^{c}=\mathbf{r});$$

$$\delta\mathbf{p}^{p}=(\mathbf{J}^{p\mathrm{T}}\mathbf{J}^{p})^{-1}(\mathbf{J}^{p\mathrm{T}}\epsilon-\mathbf{J}^{c\mathrm{T}}\mathbf{J}^{p}\delta\mathbf{p}^{c})$$

8. if$(\|\delta_{\mathbf{p}}\|\le\varepsilon_2\|\mathbf{p}\|)$
9. stop: = true;
10. else
11. $\mathbf{p}_{\mathrm{new}}:=\mathbf{p}+\delta_{\mathbf{p}};$
12. $\rho:=(\|\epsilon\|^2-\|\mathbf{x}-f(\mathbf{p}_{\mathrm{new}})\|^2)/(\delta_{\mathbf{p}}^{T}(\mu\delta_{\mathbf{p}}+\mathbf{g}));$
13. if$(\rho>0)$
14. $\mathbf{p}:=\mathbf{p}_{\mathrm{new}};$
15. $\mathbf{J}:=f'(\mathbf{p});\epsilon=\mathbf{x}-f(\mathbf{p});$
16. $\mathbf{A}:=\mathbf{J}^{\mathrm{T}}\mathbf{J};\mathbf{D}^{\mathrm{T}}\mathbf{D}=\mathrm{diag}(A);\mathbf{g}:=\mathbf{J}^{\mathrm{T}}\epsilon;$

Gain

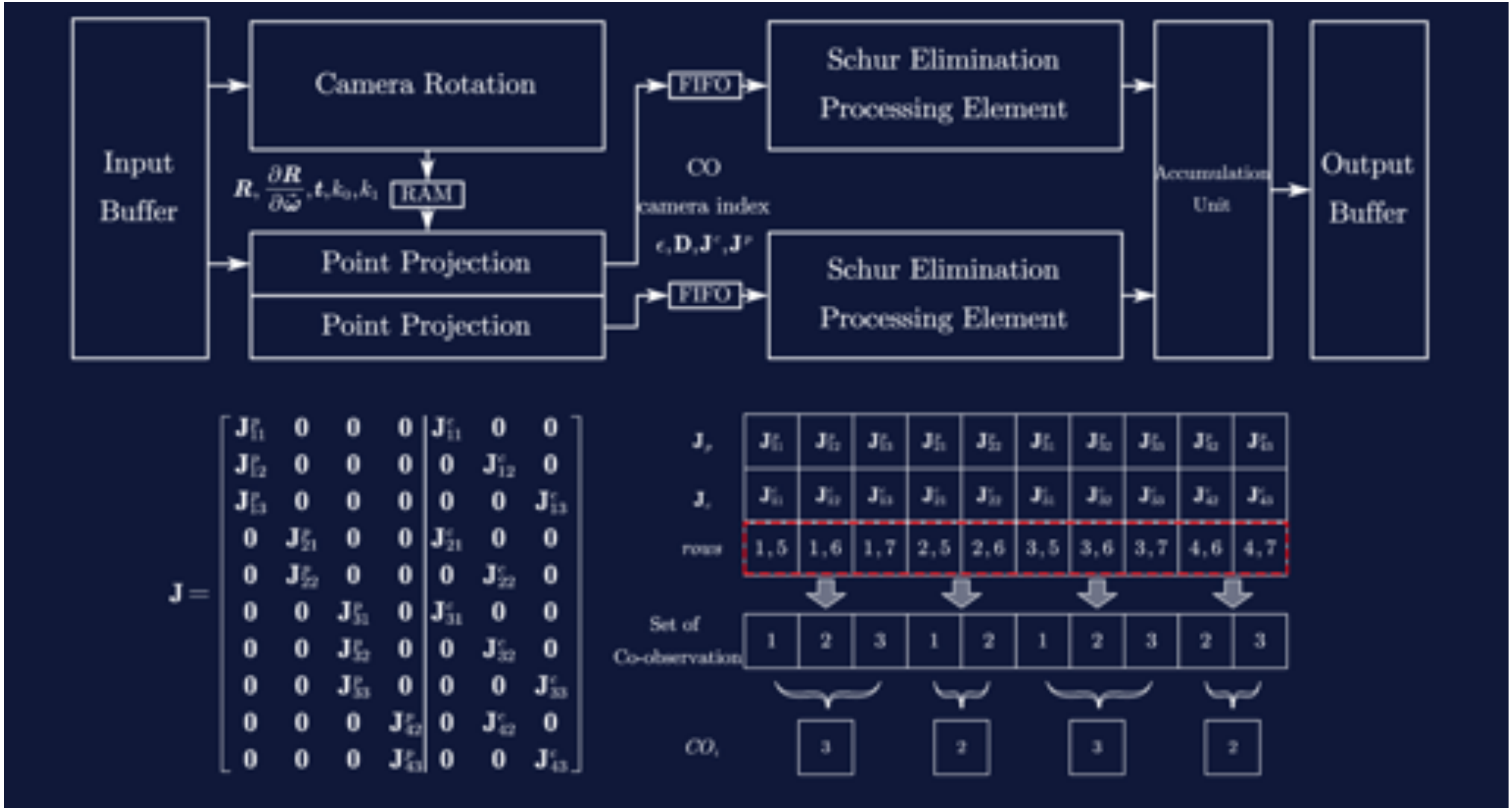
Jacobian Matrix Update

17. $\mathrm{stop}:=(\|\mathbf{g}\|_\infty\le\varepsilon_1);$
18. $\mu:=\mu^*\max\left(\frac{1}{3},1-(2\rho-1)^3\right);v:=2;$
19. else
20. $\mu:=\mu^*v;v:=2^*v;$
21. endif

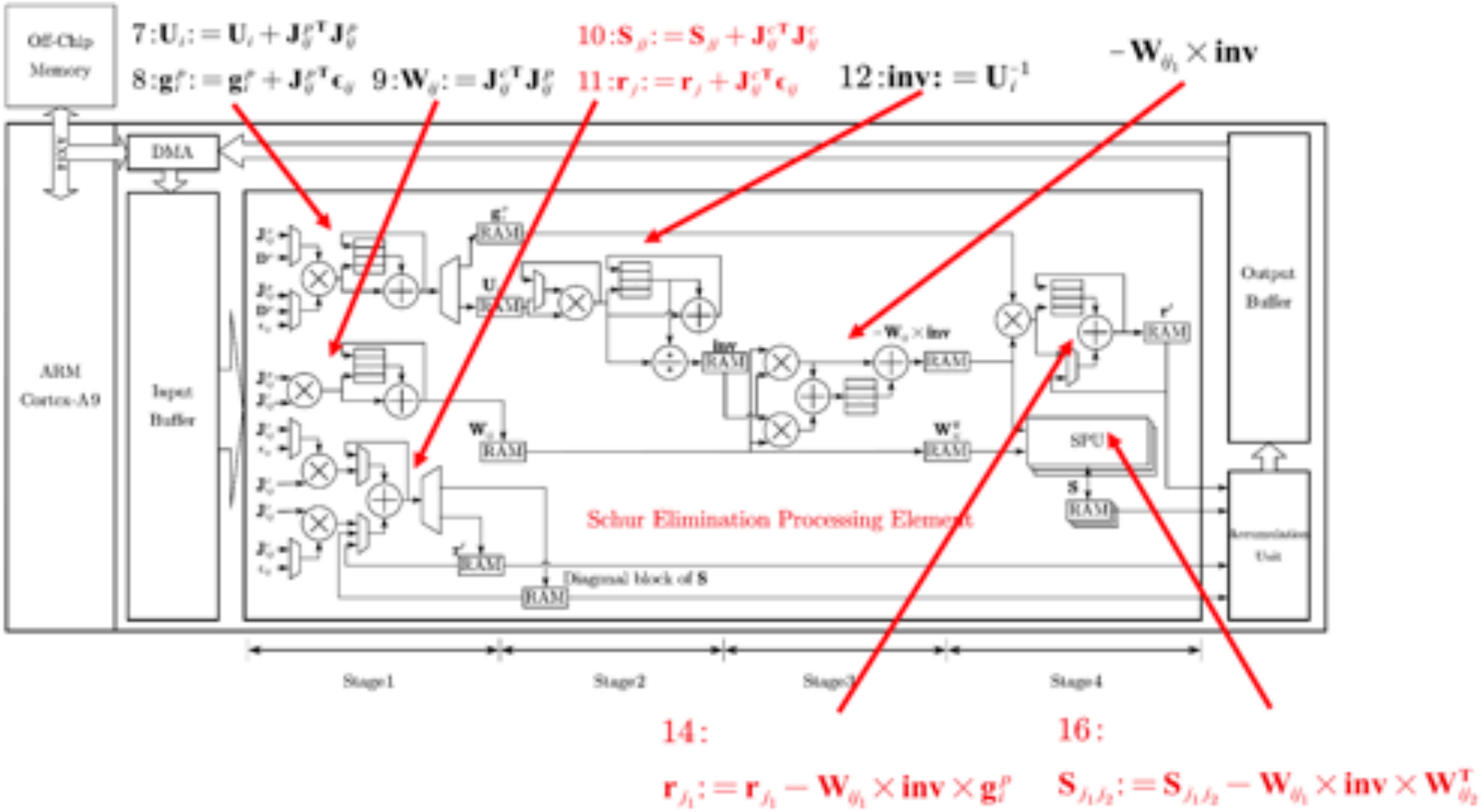
Trust Region

22. endif
23. endwhile
24. $\mathbf{p}^*:=\mathbf{p}$

Bundle Adjustment Architecture



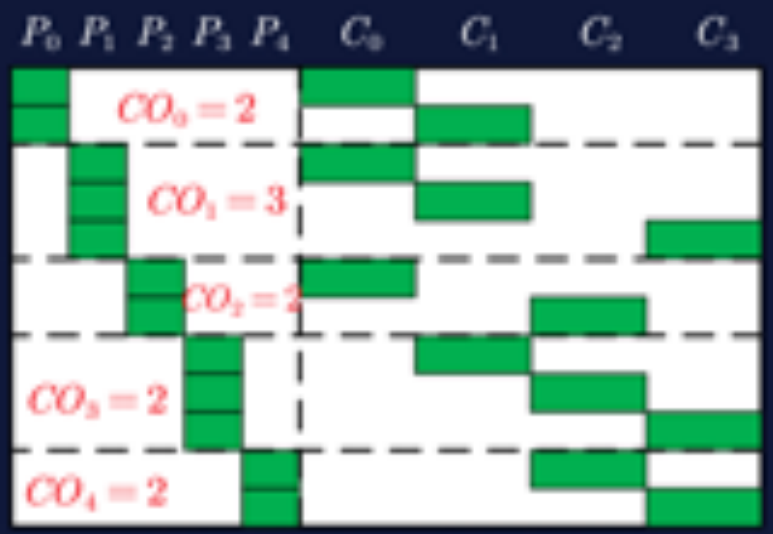
Schur Elimination Processing Element



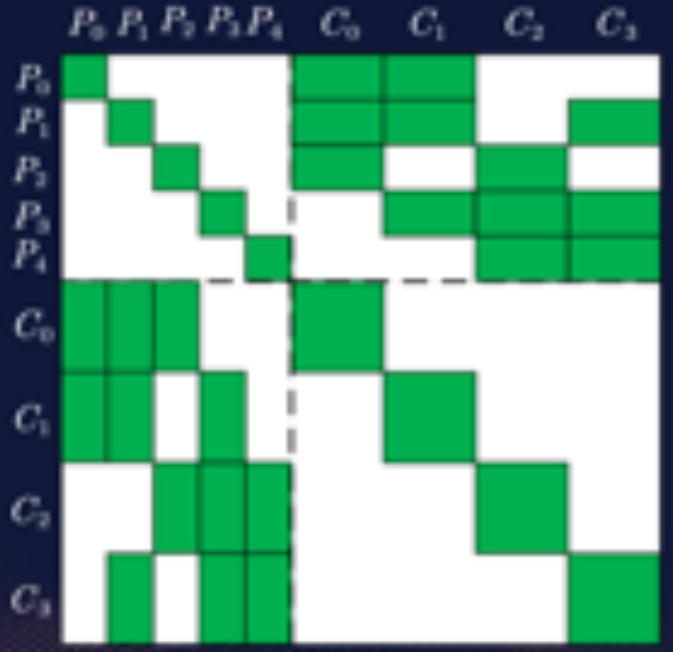
Design Challenge

- Computational complexity depends on the number of co-observations
- Trade off between latency and hardware consumption

Jacobian Matrix



$$\mathbf{J}^T\mathbf{J} + \lambda\mathbf{D}^T\mathbf{D}$$

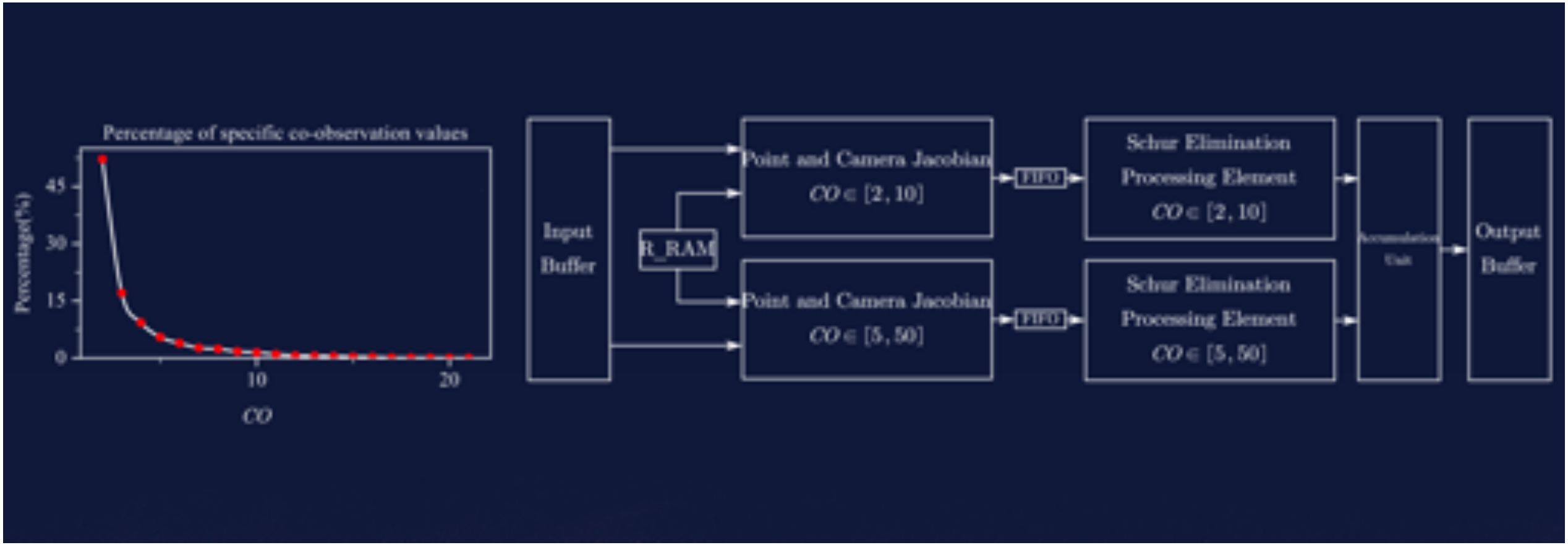


$$\begin{bmatrix} \mathbf{U} & \mathbf{W}^T \\ \mathbf{W} & \mathbf{V} \end{bmatrix} \begin{bmatrix} \delta\mathbf{p}^p \\ \delta\mathbf{p}^c \end{bmatrix} = \begin{bmatrix} \mathbf{J}^p \\ \mathbf{J}^c \end{bmatrix} \epsilon$$

$$\mathbf{V} - \mathbf{W}\mathbf{U}^{-1}\mathbf{W}^T\delta\mathbf{p}^c = (\mathbf{J}^c - \mathbf{W}\mathbf{U}^{-1}\mathbf{J}^p)\epsilon$$

$$\delta\mathbf{p}^p = \mathbf{U}^{-1}(\mathbf{J}^{pT}\epsilon - \mathbf{W}\delta\mathbf{p}^c)$$

Co-Observation Optimization Technique



Data Set and Hardware Platform

Bundle Adjustment in the Large

Data set ID in this work	#Cameras	#Points	#Observations
①	16	22106	83718
②	21	11315	36455
③	39	18060	63551
④	49	7776	31843
⑤	50	20431	73967

Xilinx Zynq-7000 SoC ZC706
with XC7Z045 FFG900-2

Jacobian and Shur Hardware Latency

Data set	FPGA (Jaco_Schur_2) @138MHz	Intel G2030 @3GHz	ARM Cortex-A9 @667MHz	Improv. vs Intel	Improv. vs ARM
①	22.993ms	102.993ms	2181.994ms	4.48x	94.90x
②	9.396ms	44.127ms	928.068ms	4.70x	98.77x
③	18.087ms	82.115ms	1680.700ms	4.54x	92.92x
④	10.948ms	42.513ms	882.640ms	3.88x	80.62x
⑤	22.152ms	97.064ms	2008.205ms	4.38x	90.65x
Average latency per iteration	16.715ms	73.763ms	1536.321ms	4.40x	91.58x

Outline

- Overview of Heavyweight Autonomous Driving
- Overview of Lightweight Autonomous Driving
- Edge Computing: Challenges and Opportunities
- Acceleration Example I: Fully Autonomous Robot Powered by a Cell Phone
- Acceleration Example II: FPGA-based ORB Feature Extraction for Real-Time Visual SLAM
- Acceleration Example III: Bundle Adjustment Acceleration on Embedded FPGAs with Co-observation Optimization
- Potential Class Projects
  - IEEE Robotics and Automation Letters (8 pages) \ IROS \ ICRA
  - IEEE Computer Architecture Letters (4 pages)
  - IEEE Embedded Systems Letters (4 pages)
  - IEEE MICRO (8 pages)

Project 1: Building Block Acceleration

IEEE ROBOTICS AND AUTOMATION LETTERS. PREPRINT VERSION. ACCEPTED JANUARY, 2018

Previous work example:

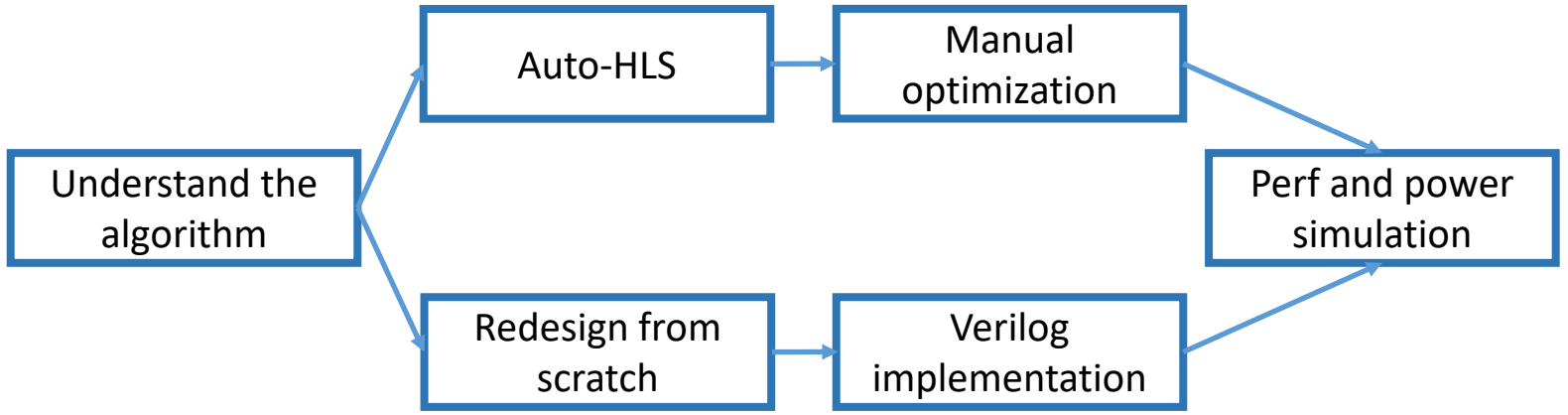
Real-Time Dense Stereo Matching with ELAS on FPGA Accelerated Embedded Devices

Oscar Rahnama[1,2], Duncan Frost[1], Ondrej Miksik[1,3] and Philip H.S. Torr[1]

Building Blocks:

- QP for Planning
- Optical Flow
- Particle Filter
- POMDP
- ICP
- E2E models

Methodologies:



Project 2: A Unified Hardware Synchronization Platform

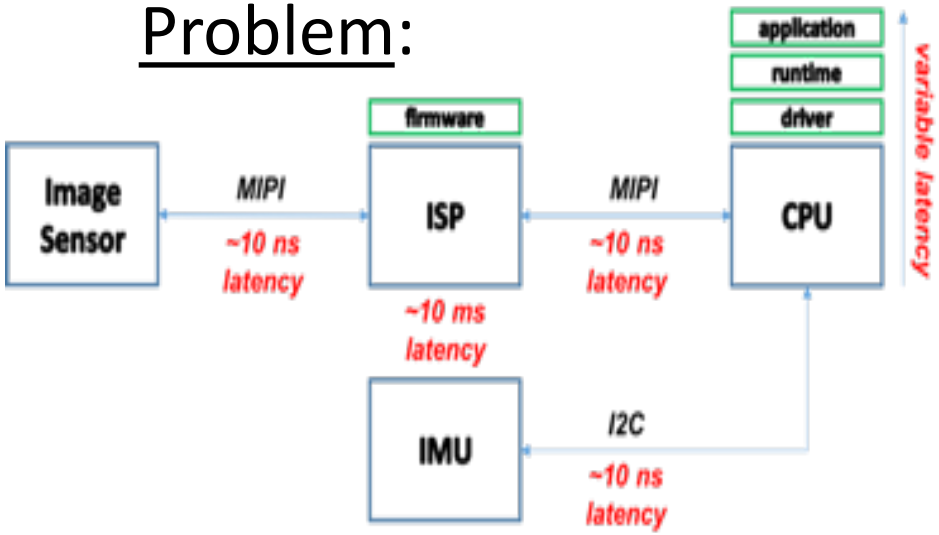
Previous work example:

A Synchronized Visual-Inertial Sensor System with FPGA Pre-Processing for Accurate Real-Time SLAM

Janosch Nikolic, Joern Rehder, Michael Burri, Pascal Gohl, Stefan Leutenegger, Paul T. Furgale and Roland Siegwart[1]

Abstract— Robust, accurate pose estimation and mapping at real-time in six dimensions is a primary need of mobile robots, in particular flying Micro Aerial Vehicles (MAVs), which still perform their impressive maneuvers mostly in controlled environments. This work presents a visual-inertial sensor unit aimed at effortless deployment on robots in order to equip them with robust real-time Simultaneous Localization and Mapping (SLAM) capabilities, and to facilitate research on this important

Problem:



Objectives:

- a hardware system to synchronize multiple sensors : stereo cameras, IMU, GNSS, radar, and LiDAR
- this system provides “clean” data for the perception and localization systems, and is the unified sensor interface for all future robots/vehicles
- need to understand how “time” works in each sensor

Project 3: Lightweight Middleware

Baseline: ROS.org

Problems:
- Very high communication overheads
- Security – lack of encryption
- Reliability – no monitor node
- Resource allocation – no limit

Solutions:
- Nanomsg as comm. backbone
- LXC for node resource constraints
- Fast encryption and decryption
- Heartbeat monitoring

Experiments:
- Launch multiple nodes: sensor node and perception node
- Stress the nodes, compare perf on ROS and the new system

Initial Results:

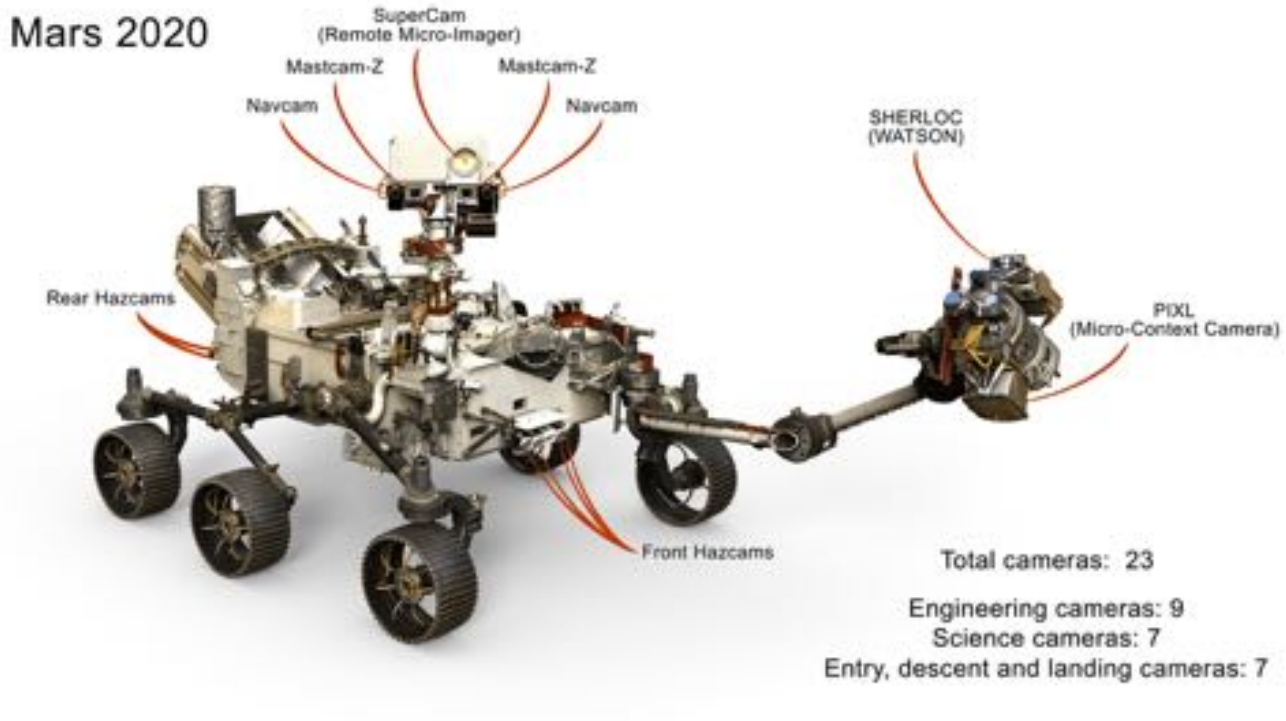
	ROS	nanomsg
mem footprint (MB)	50	0.03
throughput (MB/s)	1028	1664
latency (us)	130	90

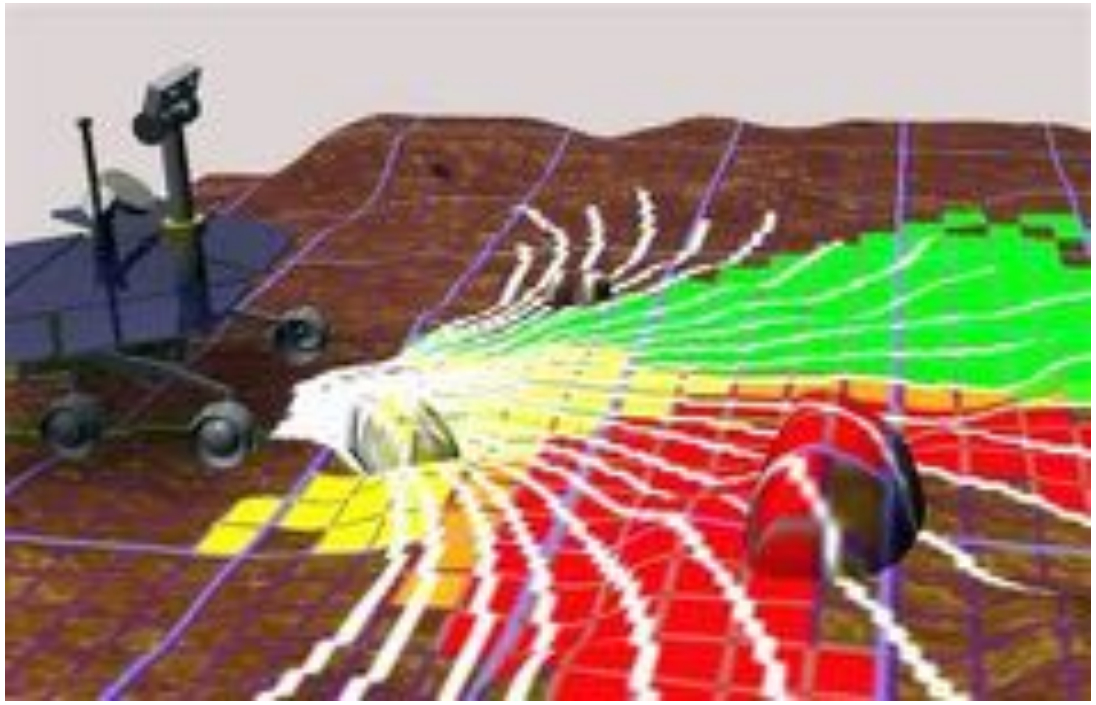
Project 4: Space Robotic Processor Design Methodology

How does a robot localize itself on Mars? (Note: no GPS)

Project 4: Space Robotic Processor Design Methodology

How does a robot localize itself on Mars? (Note: no GPS)

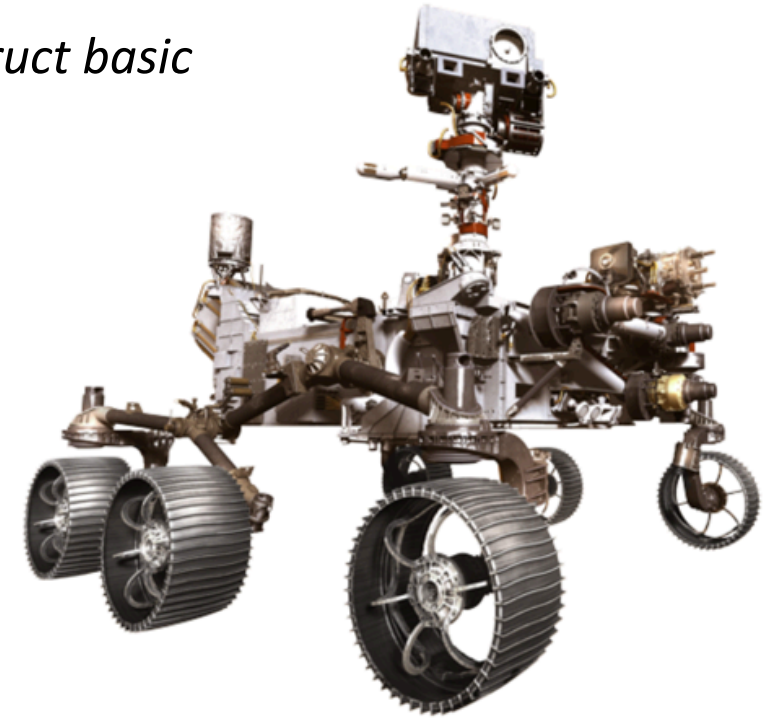




Based on variables including power levels, terrain difficulty, slippage and visibility, **the maximum terrain-traverse speed is estimated to be 200 m (660 ft) per day** by automatic navigation

Project 4: Space Robotic Processor Design Methodology

Vision: *Commercial autonomous robotic explorers will explore and construct basic infrastructure on Mars, making it habitable for mankind.*

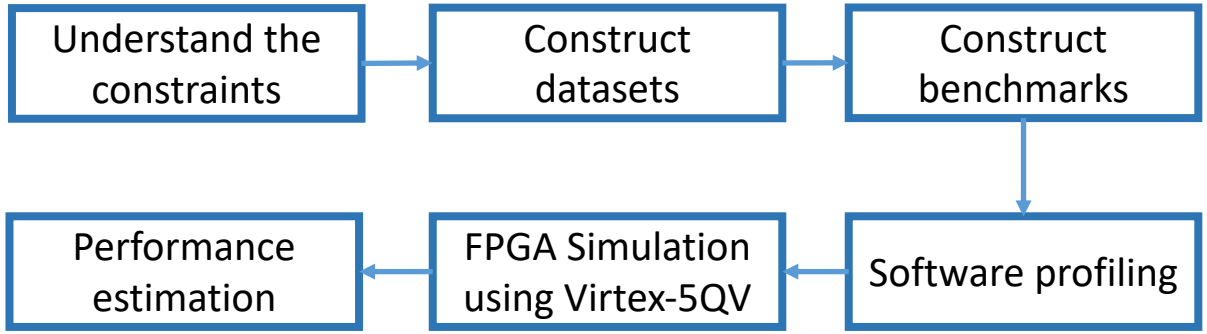


https://mars.jpl.nasa.gov/msl/mission/rover/

Objectives:

- Develop powerful computing systems for space robotic applications
- Design under constraints: temperature range, radiation-hardened
- Develop datasets and benchmarks to evaluate processor designs
- Initial design using RISC-V

Methodology:



https://www.xilinx.com/products/silicon-devices/fpga/virtex-5qv.html

https://www.osti.gov/servlets/purl/1458107 Space-grade Virtex-5QV FPGA

PDS Cartography and Imaging Sciences Node
Jet Propulsion Laboratory U.S. Geological Survey

https://pds-imaging.jpl.nasa.gov/

Feel free to contact me

shaoshanliu@hks.harvard.edu