Navion: An Energy-Efficient Visual-Inertial Odometry Accelerator for Micro Robotics and Beyond

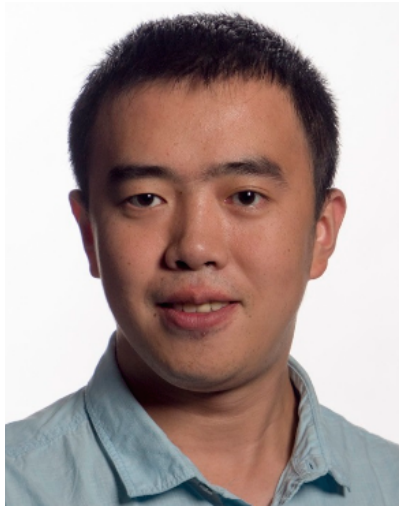
Amr Suleiman, Zhengdong Zhang, Luca Carlone,
Sertac Karaman, and Vivienne Sze

Massachusetts Institute of Technology

http://navion.mit.edu/

Contributors

Amr Suleiman
PhD Graduate in EECS



Zhengdong Zhang
PhD Candidate in EECS

Luca Carlone
Professor in AeroAstro

Sertac Karaman
Professor in AeroAstro

Vivienne Sze
Professor in EECS

Motivation: Autonomous Navigation

Self Driving Cars

UAVs: Unmanned Aerial Vehicles

Robots

[images] Electrek, Amazon, Knightscope, Boston Dynamics

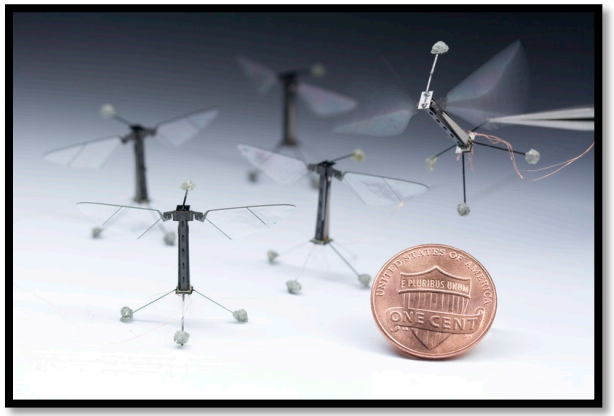
Motivation: Miniaturized Robots

Very small form factor

Micro

Nano

Pico?



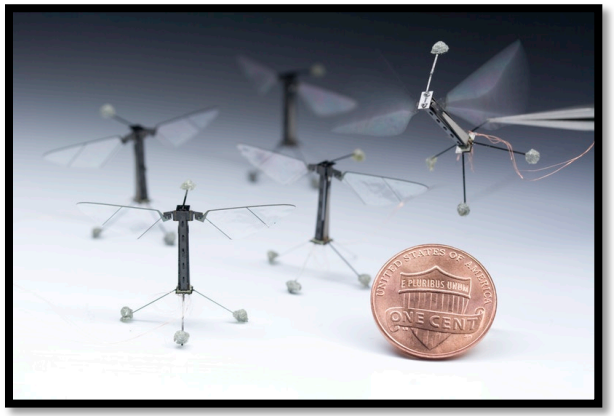
Motivation: Miniaturized Robots

Very small form factor

Micro

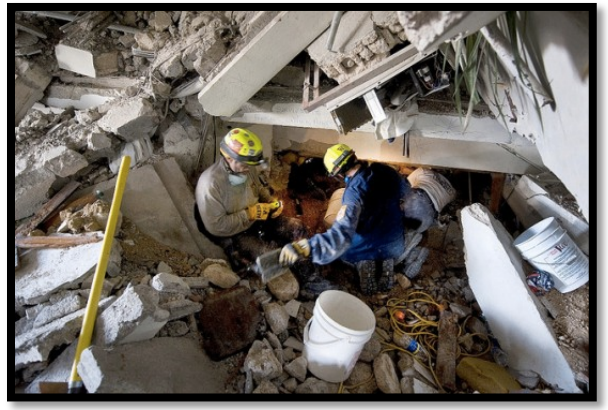
Nano

Pico?

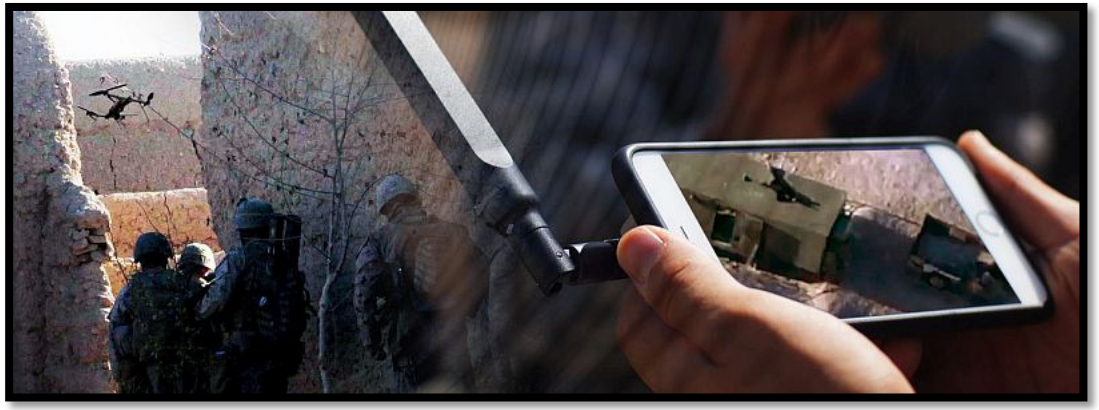


Applications

Search & Rescue



Surveillance



[images] IEEE Spectrum, MIT news, Harvard, Printerest, Intelligent Aerospace

How Does Autonomous Navigation Work?

Perception

Motion Planning

Control

How Does Autonomous Navigation Work?

Perception

Motion Planning

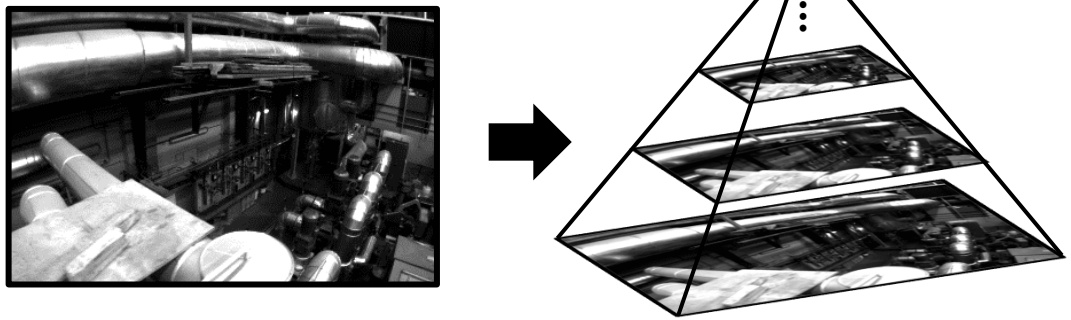
Control

Perception is the computation bottleneck

[Kanellakis et al., JIRS 2017]

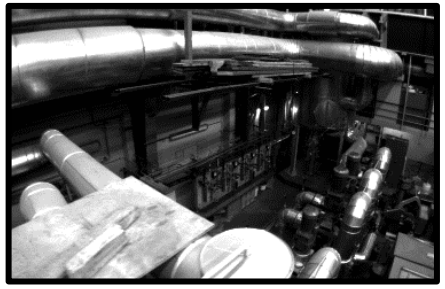
Challenges: High Dimensionality

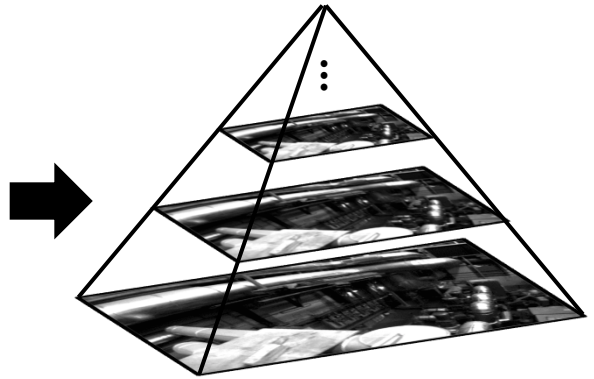
- **Large amount of data**
  - Sensors data: High resolution & frame rates
  - Data expansion: Image pyramid



Challenges: High Dimensionality

- **Large amount of data**
  - Sensors data: High resolution & frame rates
  - Data expansion: Image pyramid





- **Growing map size**

[T. Pire et al., 2017]

Challenges: Low Power Budget

Big battery

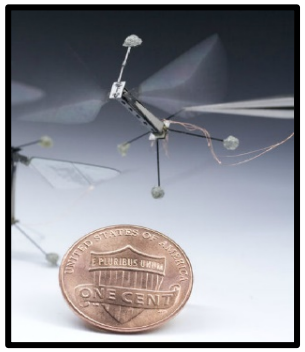
Mobile CPU, GPU

Challenges: Low Power Budget

Big battery

Mobile CPU, GPU

For example:

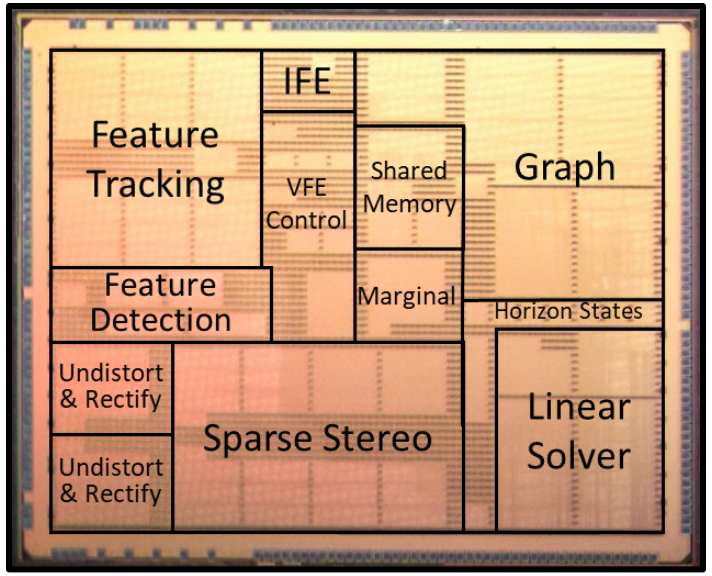


Insect-scale UAV (100mg)

Lifting	Sensing	CPU, GPU
100 mW	100 mW	10 - 100 W

[R.J. Wood et al., IJRR 2012]

Navion: Energy-Efficient Visual-Inertial Odometry



- Energy-efficient & real-time localization and mapping
- Real-time performance of 20 fps at 2mW
- Peak performance of up to 171 fps at 24 mW

Outline

- Localization & Mapping: Visual-Inertial Odometry (VIO)
- Chip Architecture
- Main Contributions
- Chip Specifications and Comparisons
- Summary

Localization and Mapping Using VIO

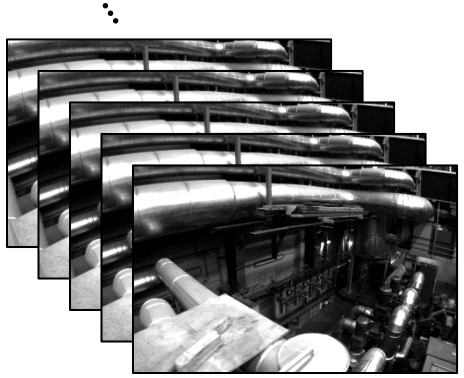
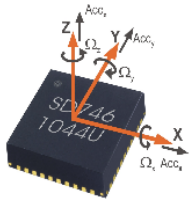


Image sequence

IMU

Inertial Measurement Unit



Visual-Inertial Odometry (VIO)

Localization and Mapping Using VIO

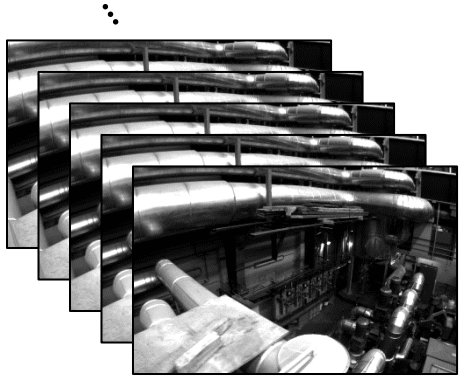
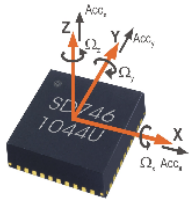


Image sequence

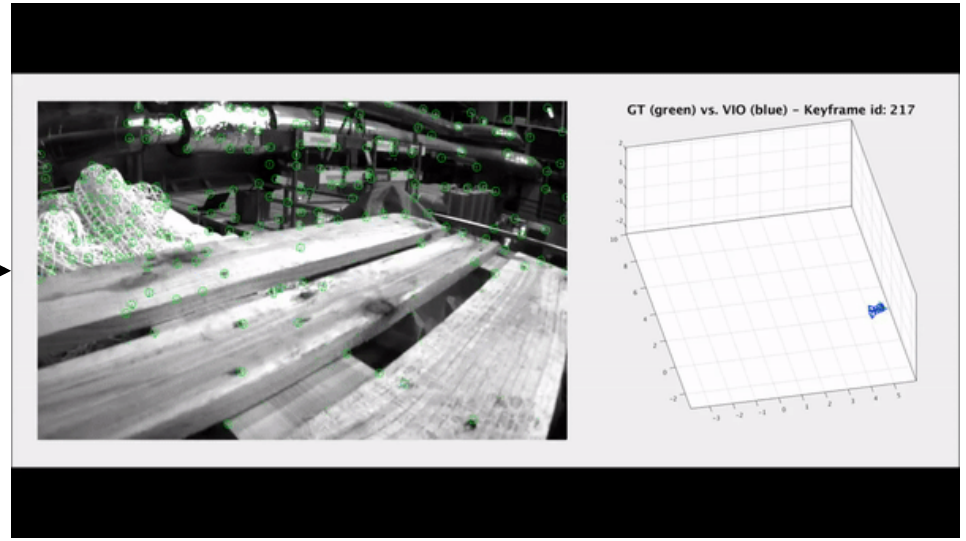
IMU

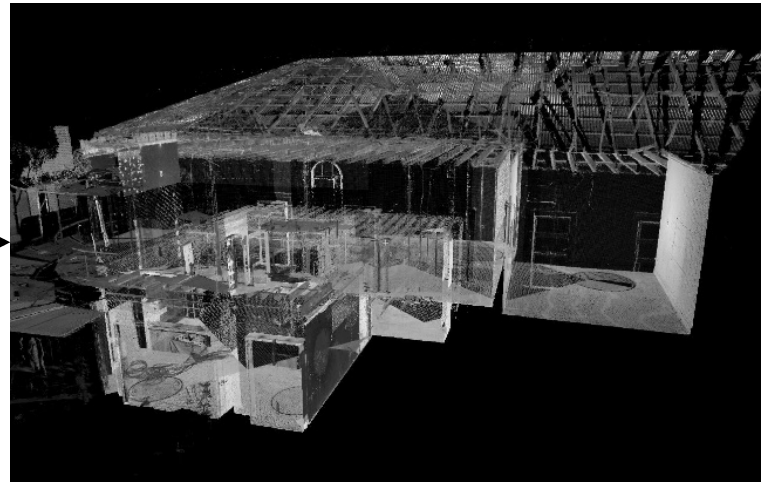
Inertial Measurement Unit



Visual-Inertial Odometry (VIO)

Localization





Mapping

Localization and Mapping Using VIO

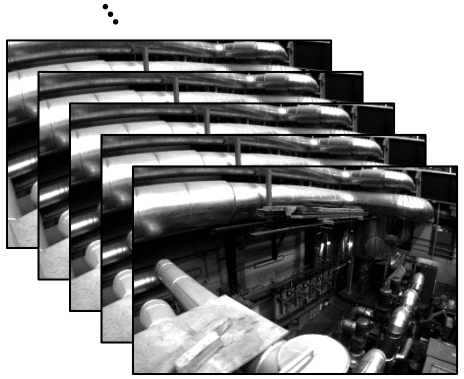
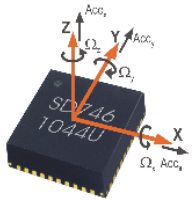


Image sequence

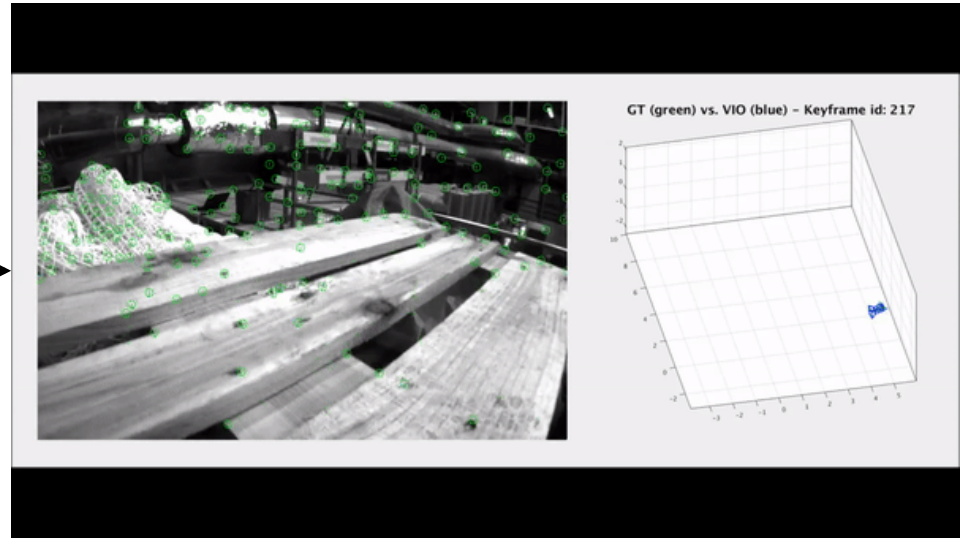
IMU

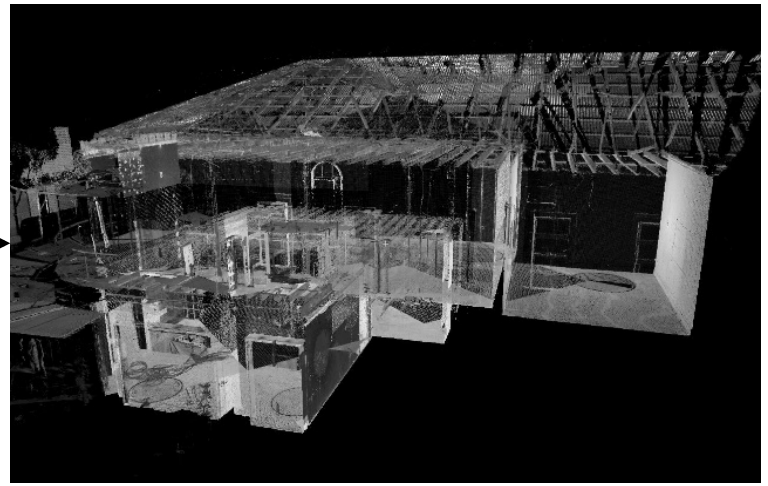
Inertial Measurement Unit



Visual-Inertial Odometry (VIO)

Localization

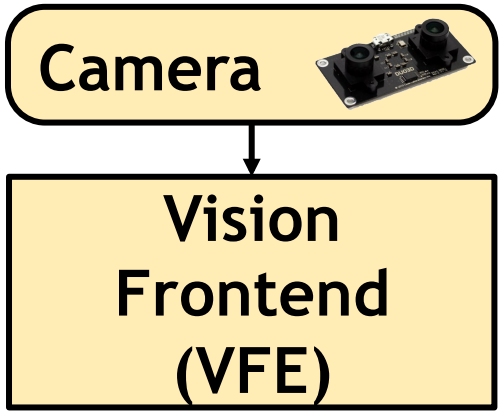


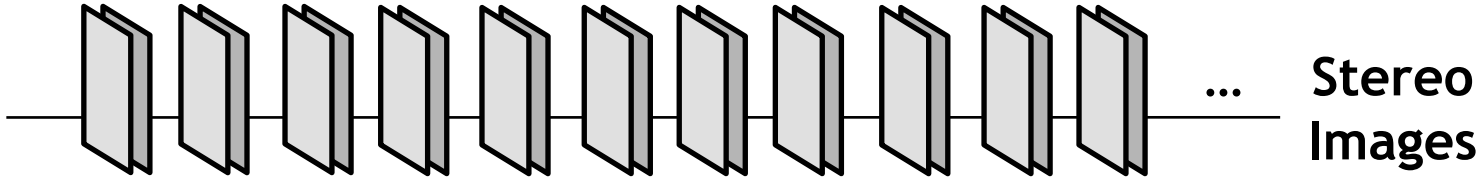


Mapping

Subset of SLAM algorithm
(Simultaneous Localization And Mapping)

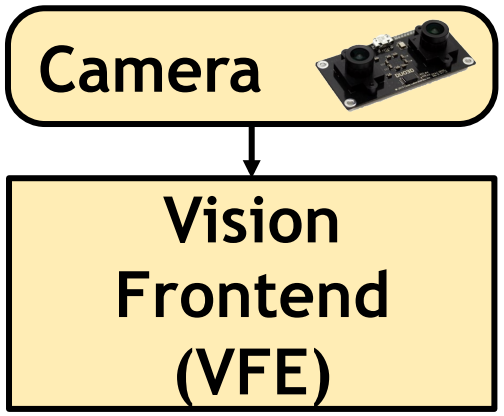
VIO: Frontend

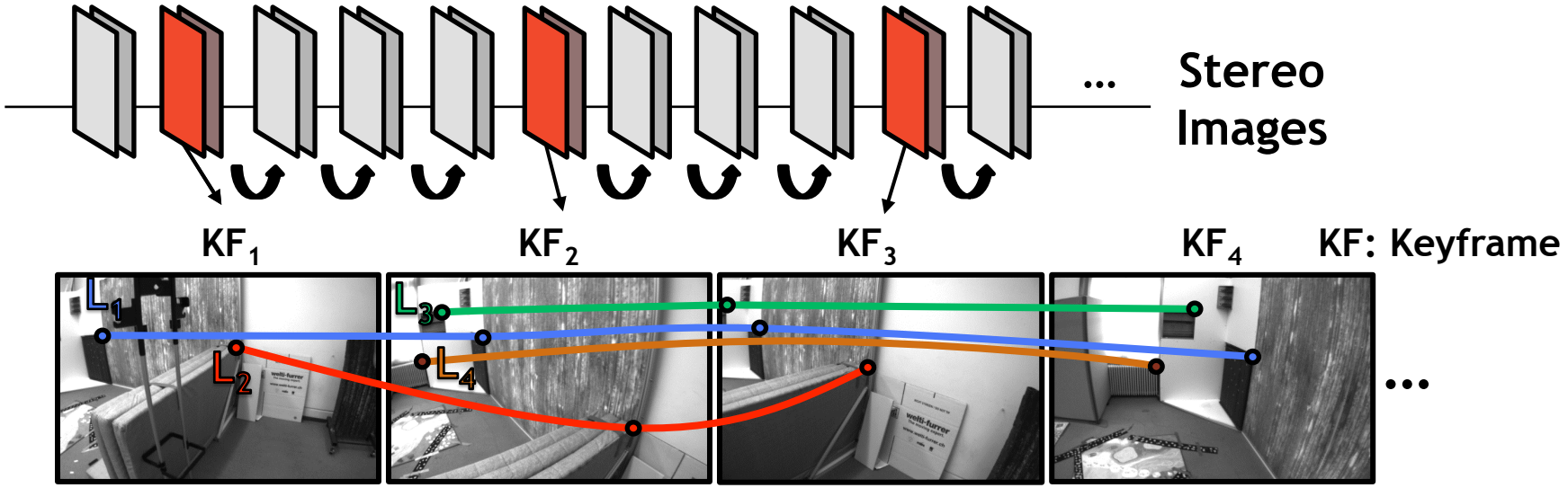




Process mono/stereo Images

VIO: Frontend

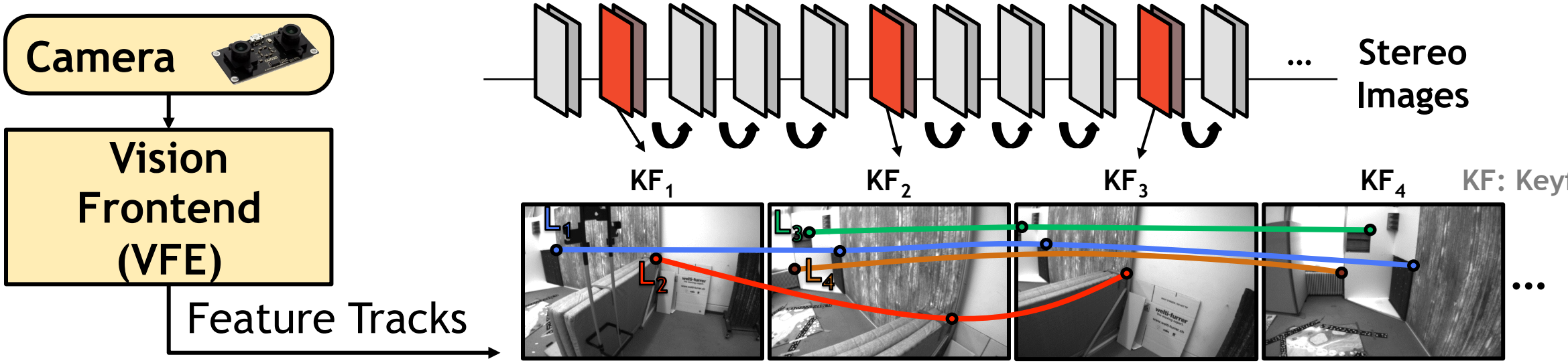




Process mono/stereo Images

- Detect & track features (L_i)

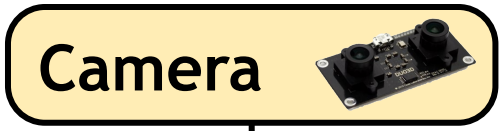
VIO: Frontend



Process mono/stereo Images

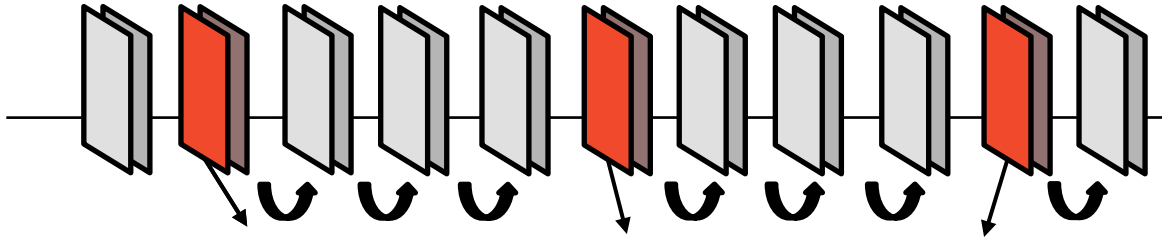
- Detect & track features (L_i)
- Generate ***Feature Tracks*** → (keyframe IDs & feature coordinates)

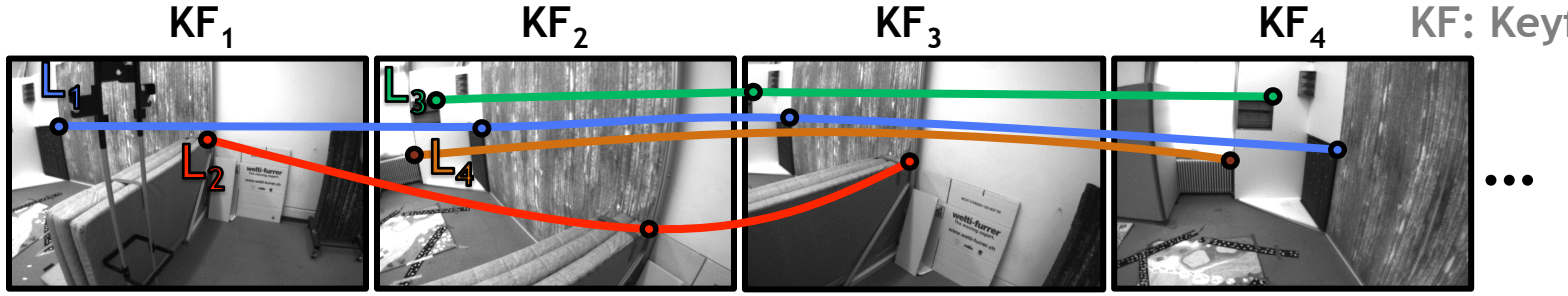
VIO: Frontend



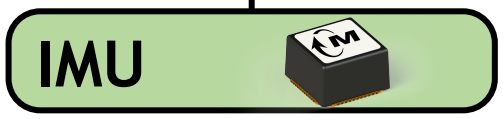
Vision Frontend (VFE)

Feature Tracks





IMU Frontend (IFE)



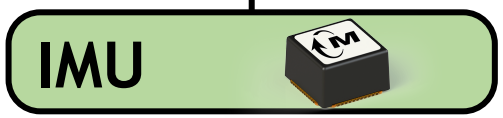
VIO: Frontend

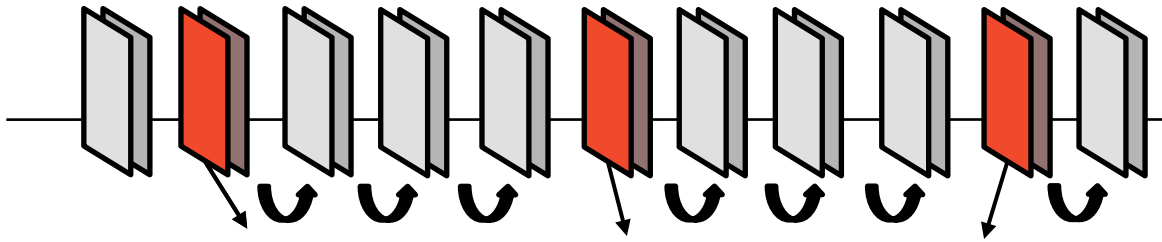
Vision Frontend (VFE)

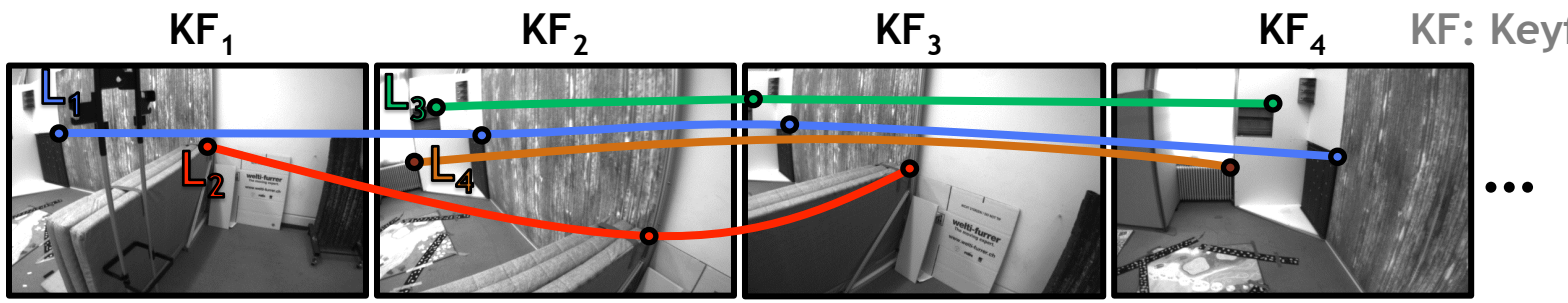
Feature Tracks

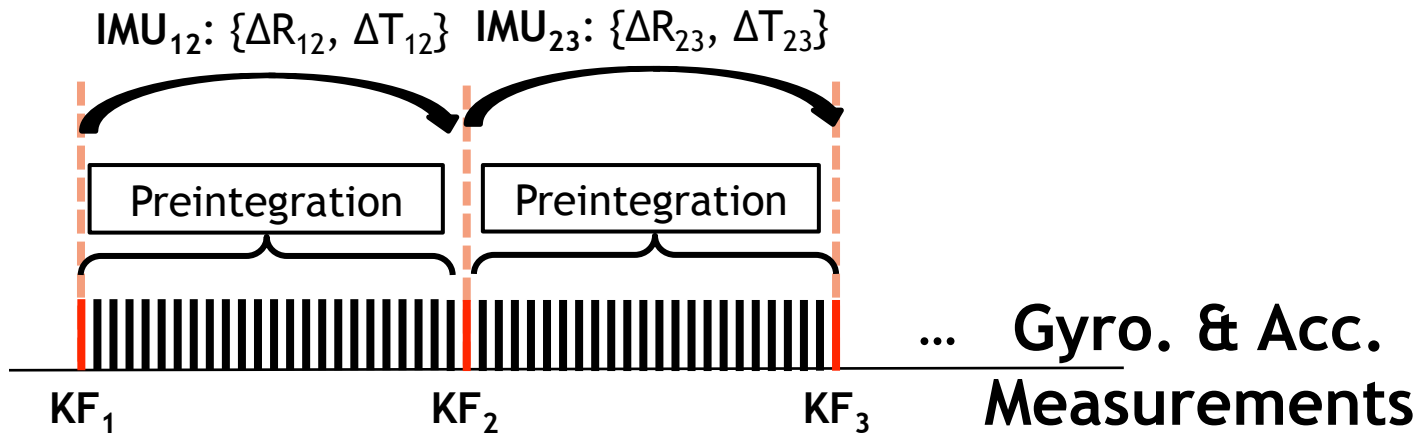
Estimated States

IMU Frontend (IFE)









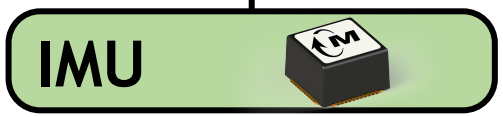
VIO: Frontend

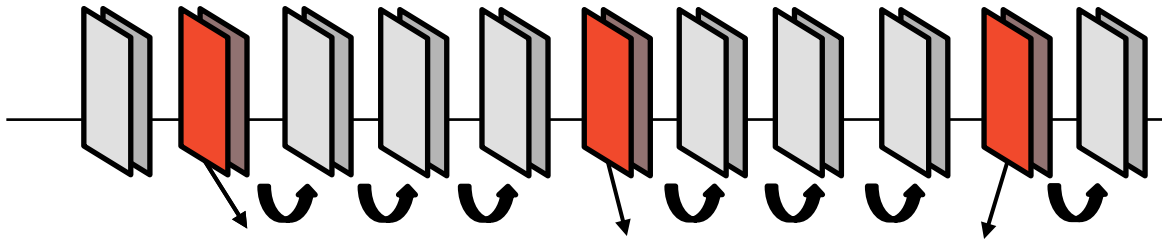
Vision Frontend (VFE)

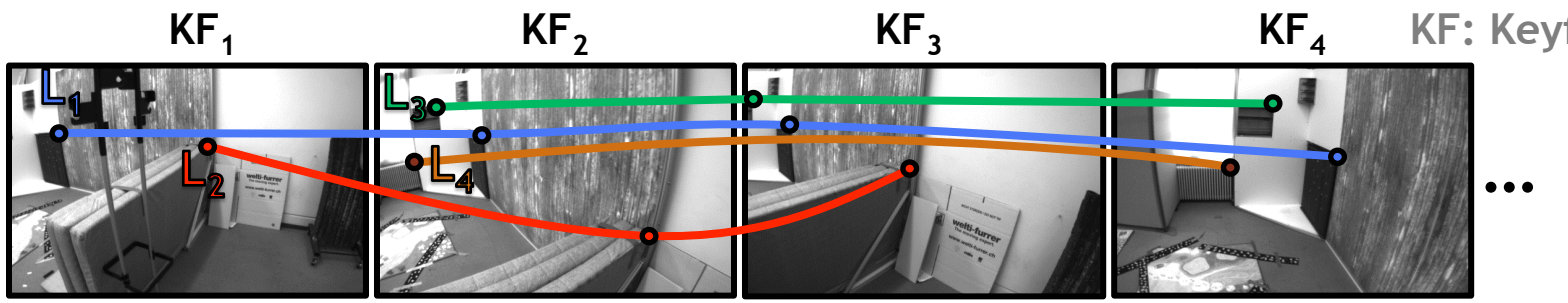
Feature Tracks

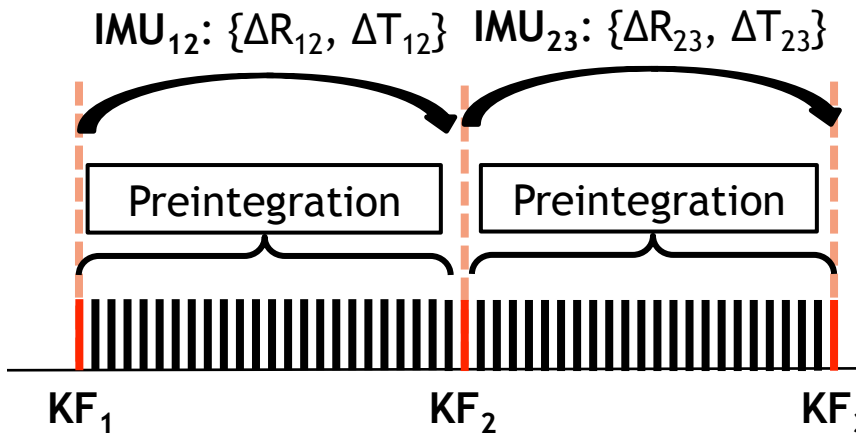
Estimated States

IMU Frontend (IFE)









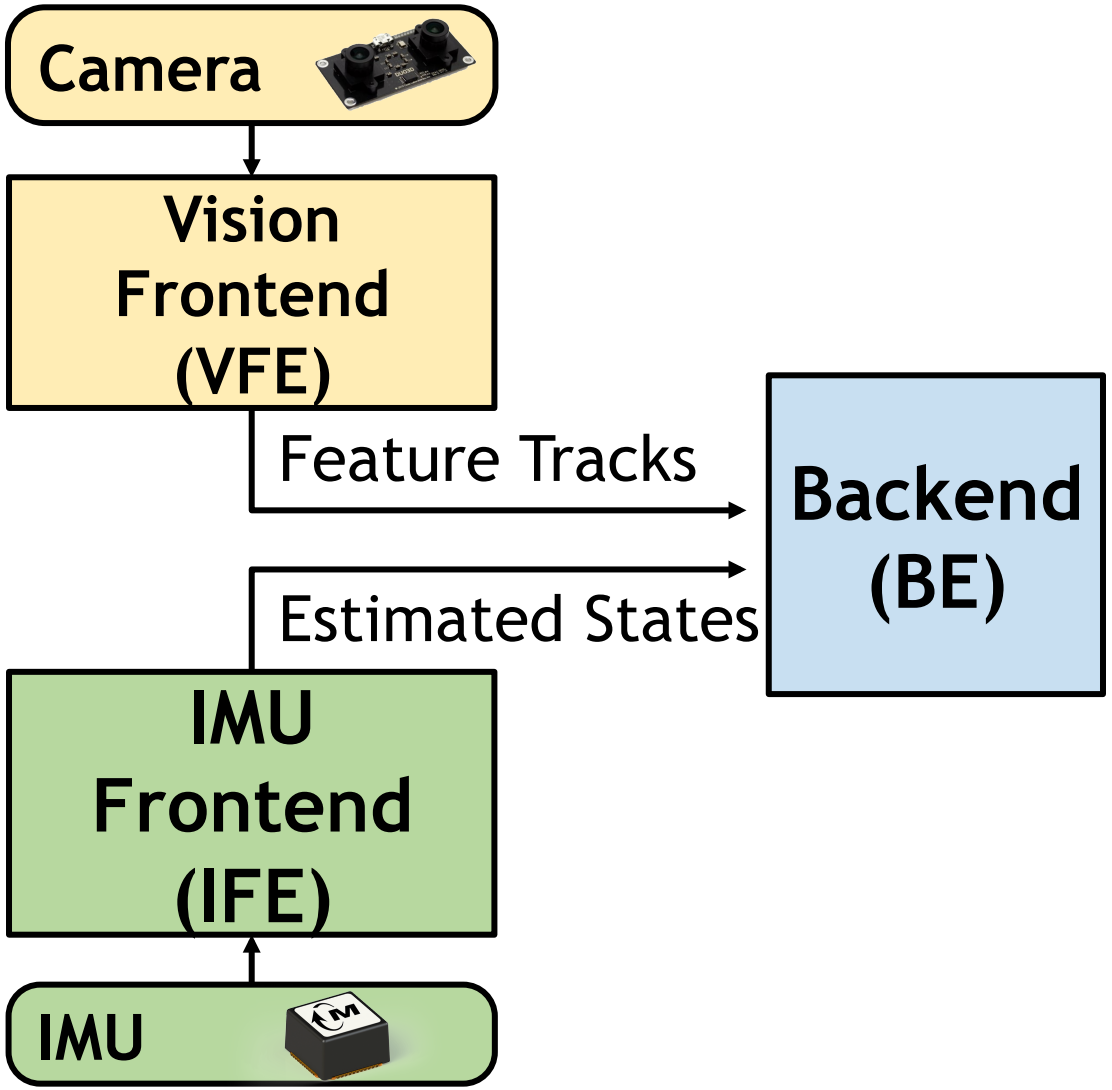
... Gyro. & Acc. Measurements

State:

Pose (Rotation **R**)

Location (Translation **T**)

VIO: Backend


Camera
Vision Frontend (VFE)
Feature Tracks
Backend (BE)
Estimated States
IMU Frontend (IFE)
IMU

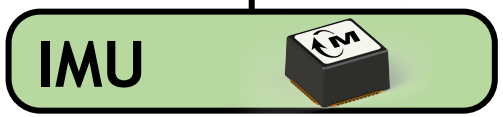
VIO: Backend

Vision Frontend (VFE)

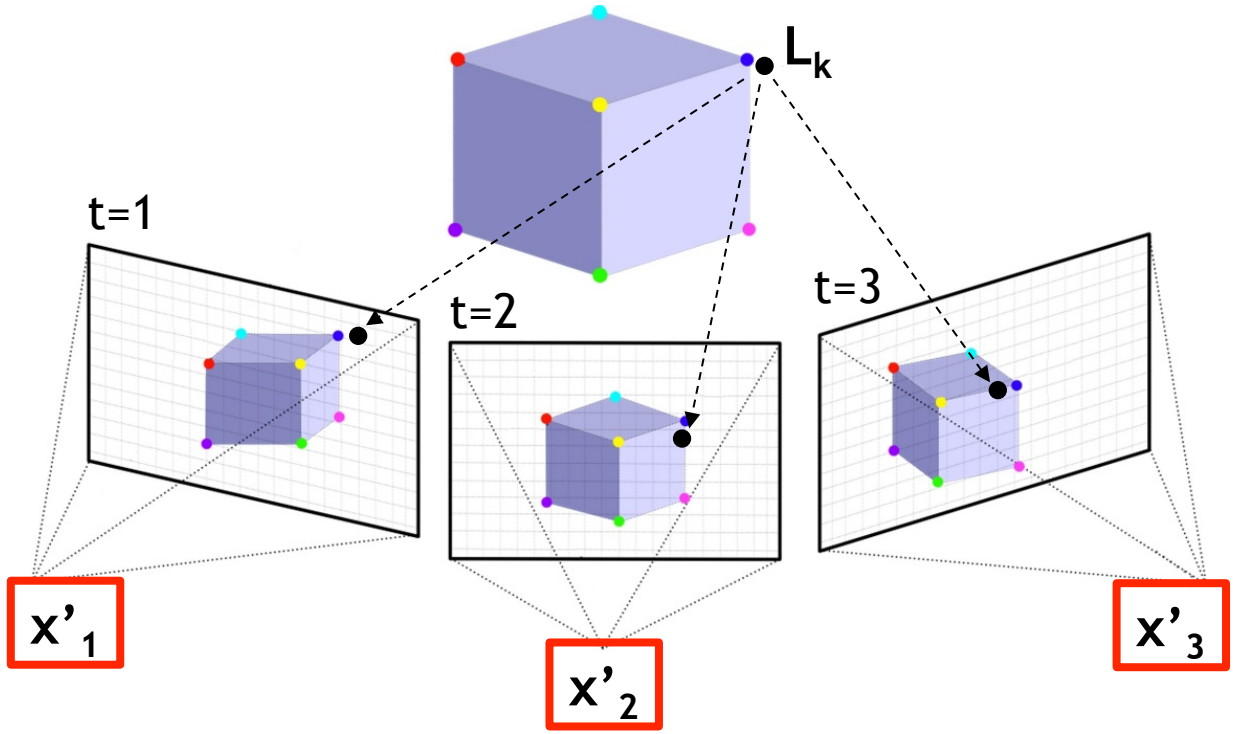
Feature Tracks

Estimated States

IMU Frontend (IFE)

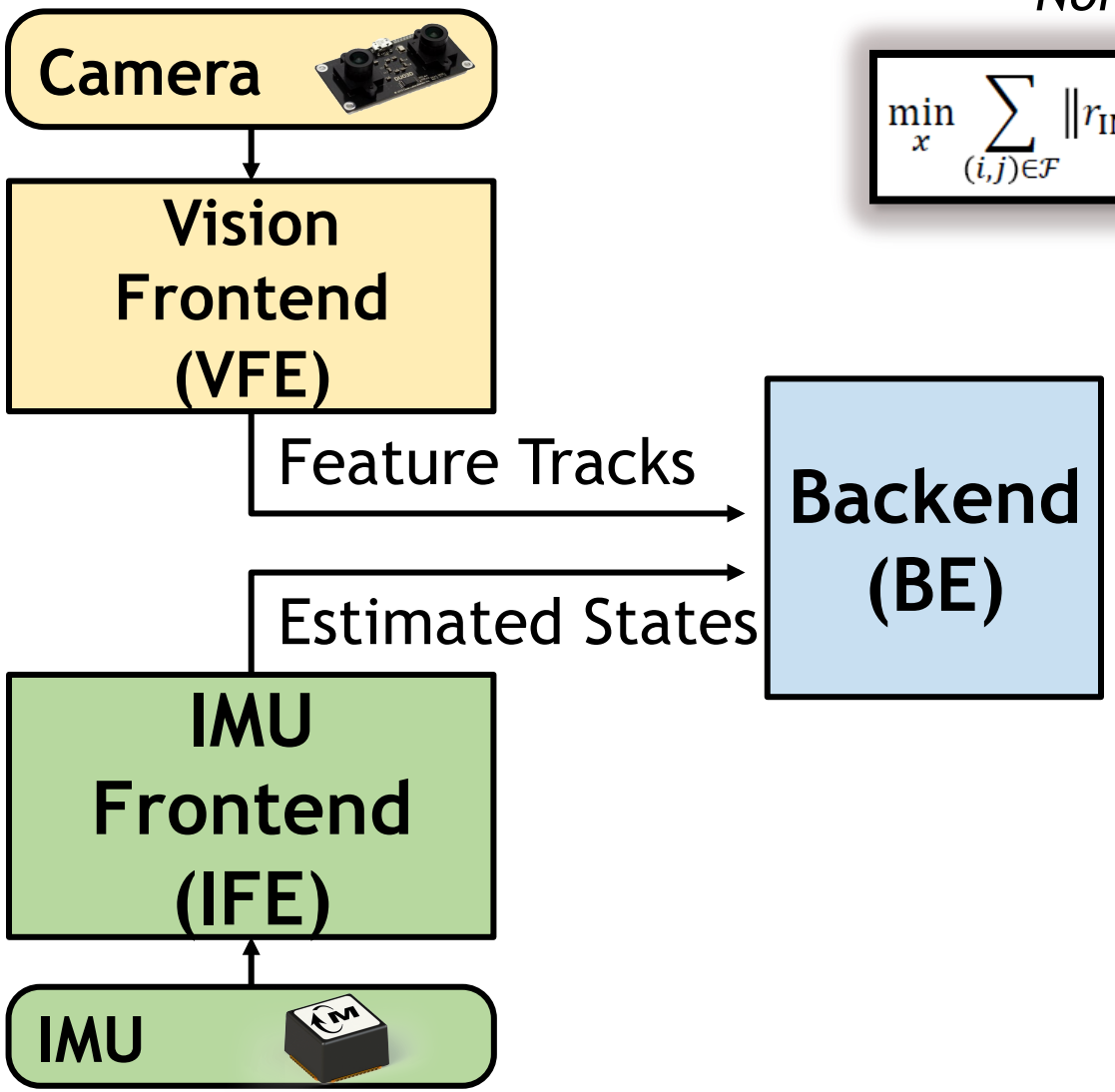


Backend (BE)



Update states (x_i) to minimize inconsistencies between measurements across time

VIO: Backend

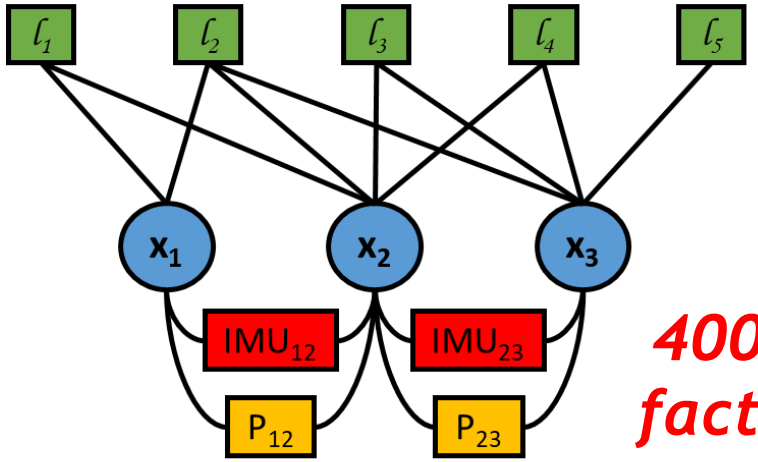


Non-linear least squares factor graph optimization

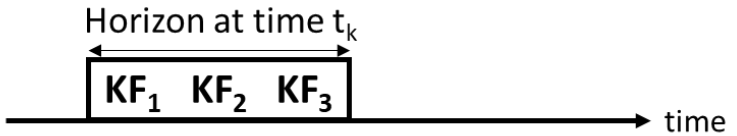
$$\min_{x} \sum_{(i,j)\in\mathcal{F}} \left\| r_{\text{IMU}}\left(x, \Delta\tilde{R}_{ij}, \Delta\tilde{p}_{ij}, \Delta\tilde{v}_{ij}\right) \right\|^2 + \sum_{k\in\mathcal{L}} \sum_{i\in\mathcal{F}_k} \left\| r_{\text{CAM}}\left(x, l_k, u_{ik}^l, u_{ik}^r\right) \right\|^2 + \left\| r_{\text{PRIOR}}(x) \right\|^2$$

IMU Factors *Vision Factors* *Other Factors*

Factor Graph



4000+ factors



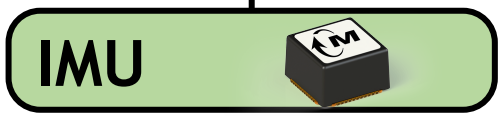
VIO: Backend

Vision Frontend (VFE)

Feature Tracks

Estimated States

IMU Frontend (IFE)



Backend (BE)

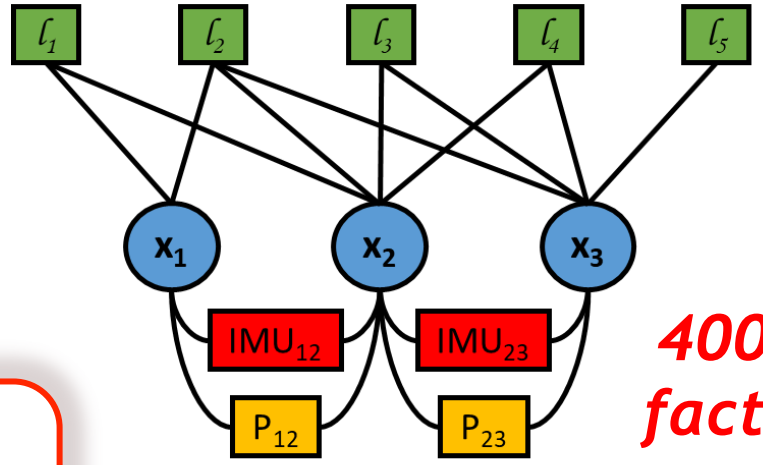
Updated States (x_i) & Sparse 3D Map

Non-linear least squares factor graph optimization

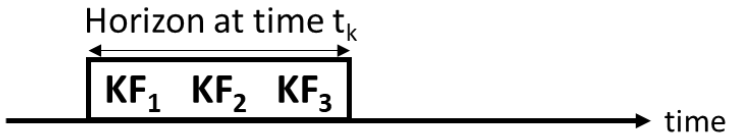
$$\min_{x} \sum_{(i,j)\in\mathcal{F}} \left\| r_{\text{IMU}}(x, \Delta\tilde{R}_{ij}, \Delta\tilde{p}_{ij}, \Delta\tilde{v}_{ij}) \right\|^2 + \sum_{k\in\mathcal{L}} \sum_{i\in\mathcal{F}_k} \left\| r_{\text{CAM}}(x, l_k, u_{ik}^l, u_{ik}^r) \right\|^2 + \left\| r_{\text{PRIOR}}(x) \right\|^2$$

IMU Factors *Vision Factors* *Other Factors*

Factor Graph



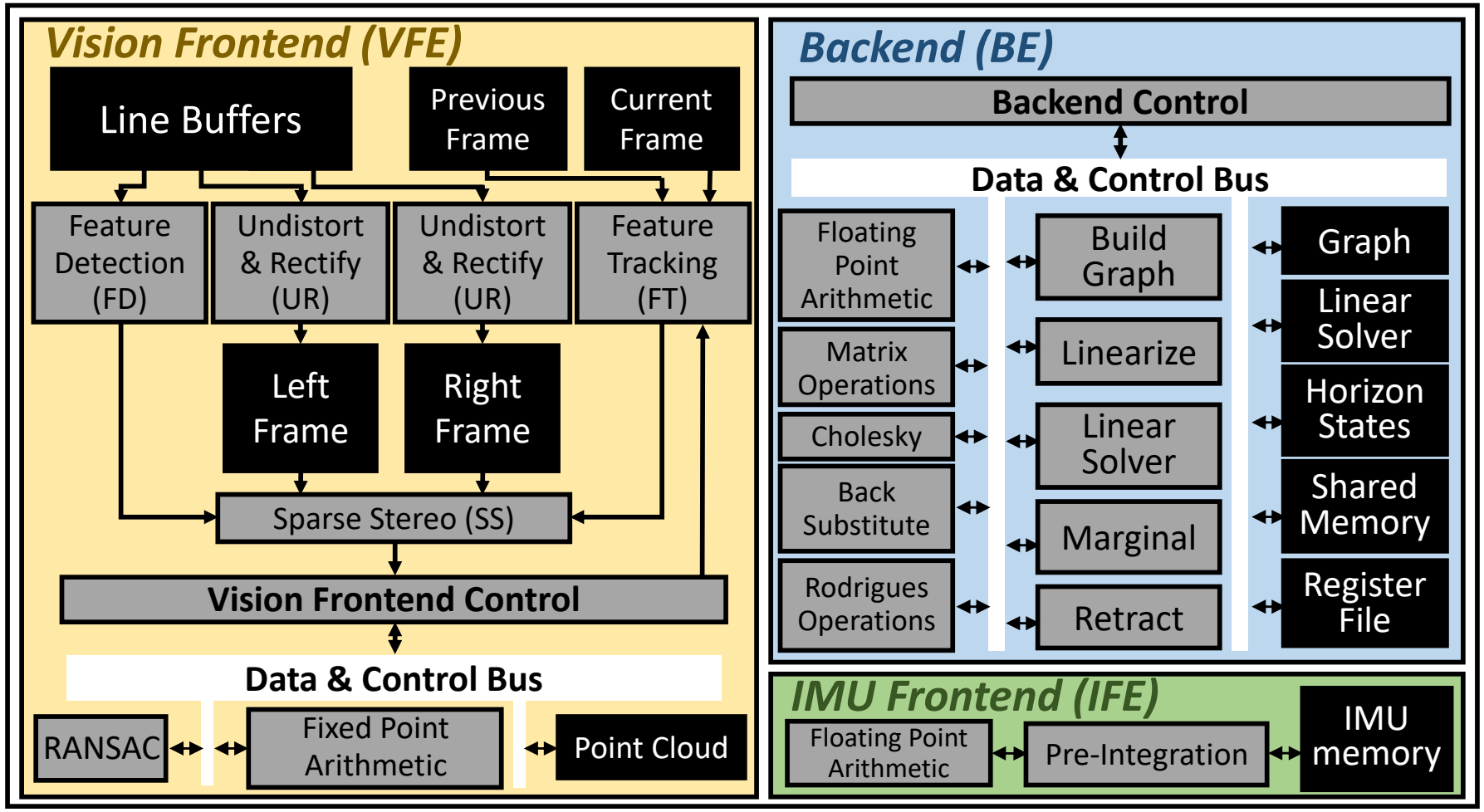
4000+ factors



Outline

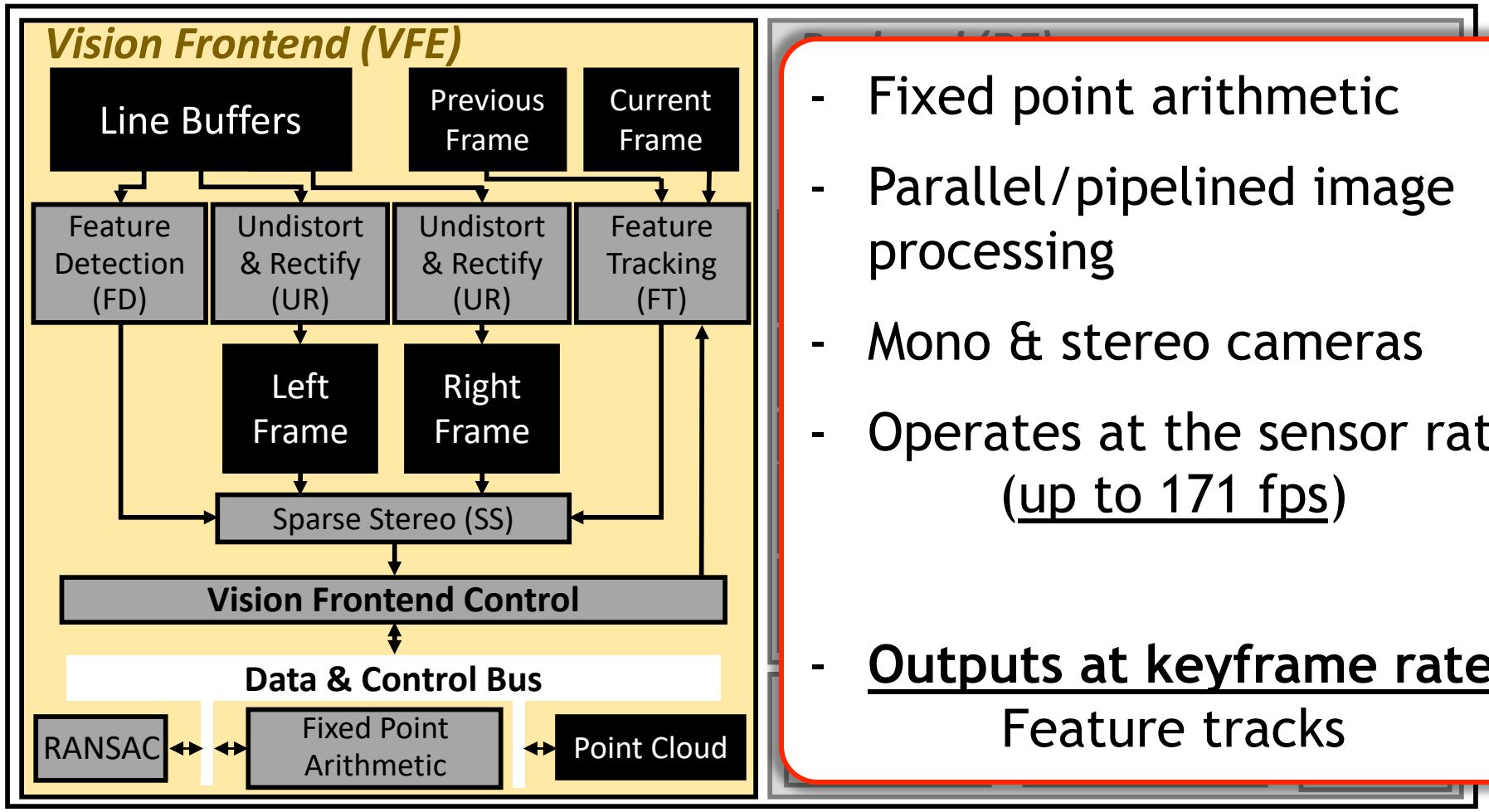
- Localization & Mapping: Visual-Inertial Odometry (VIO)
- Chip Architecture
- Main Contributions
- Chip Specifications and Comparisons
- Summary

Navion Chip Architecture



Navion is a <u>fully integrated</u> system: No off-chip storage or processing

VFE: All Image Processing

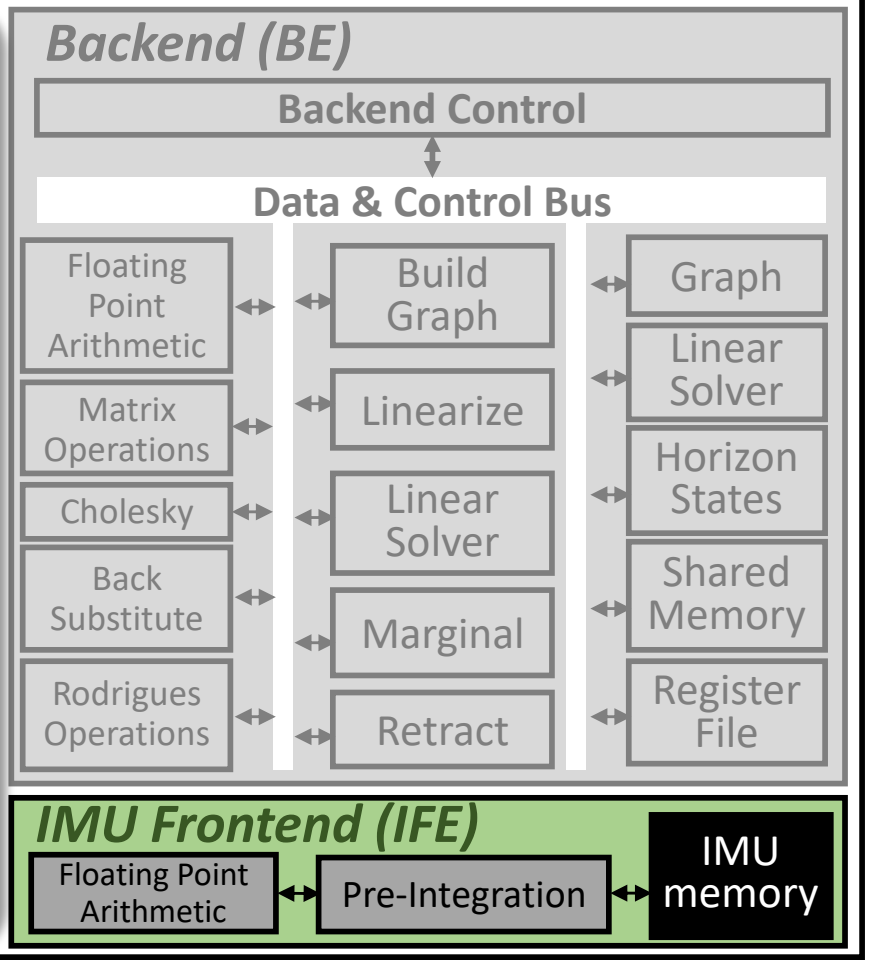


- Fixed point arithmetic
- Parallel/pipelined image processing
- Mono & stereo cameras
- Operates at the sensor rate (up to 171 fps)

- **Outputs at keyframe rate:** Feature tracks

IFE: IMU Preintegration

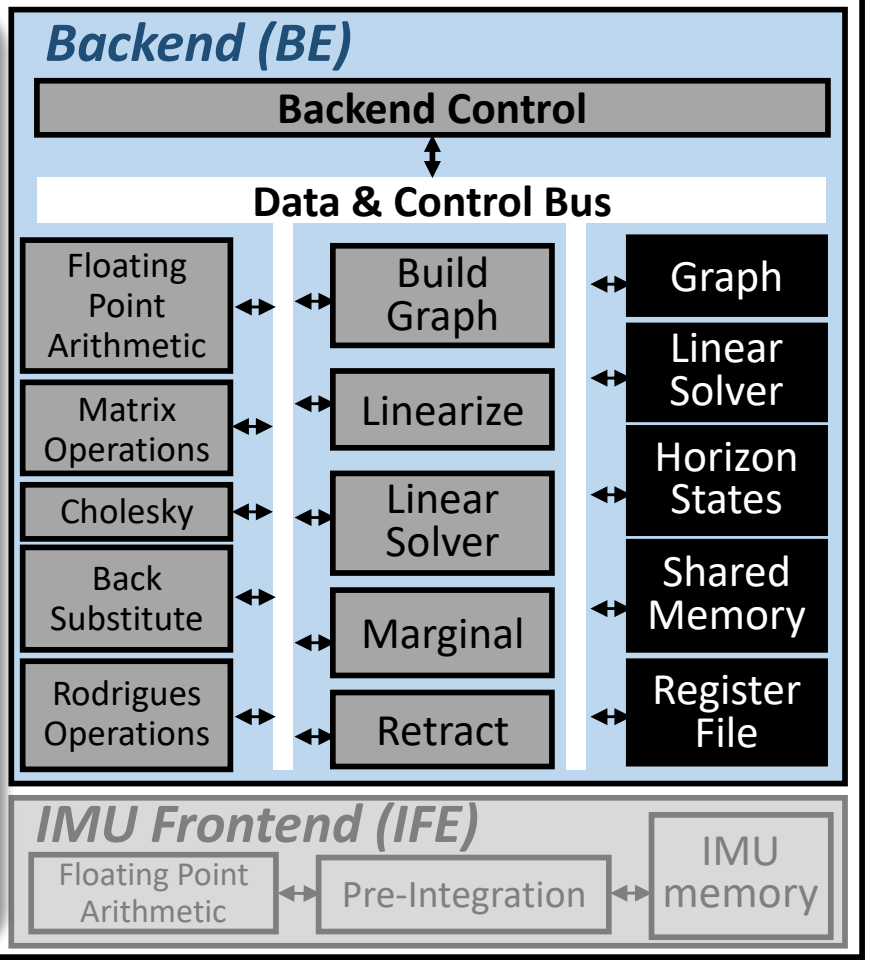
- Double precision arithmetic
- Low cost: 2.4% area & 1.2% power
- Operates at the sensor rate (up to 52 kHz)
- **Outputs at keyframe rate:** Estimated state



BE: Fusing Sensors Data

- Double precision arithmetic
- Complex Finite State Machine (FSM)
- Operates at the keyframe rate
  (up to 90 fps)

- **Outputs at keyframe rate:**
  Updated state & 3D map



VIO Full Integration Challenges

- Vision Frontend (VFE)
  - Heterogeneous computation modules
    - Feature detection
    - Feature tracking
    - Stereo matching
    - Outliers rejection using RANSAC
    - …

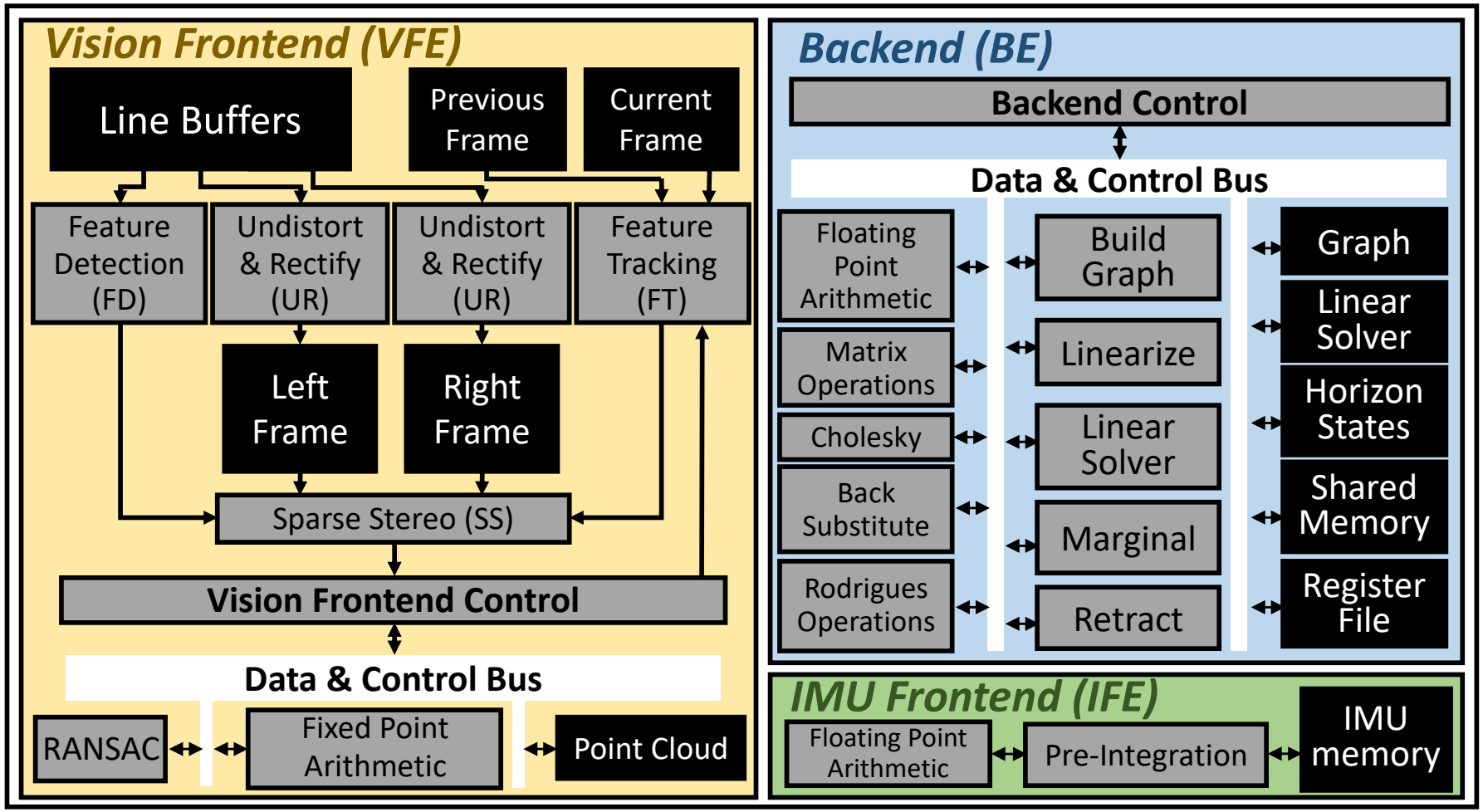
VIO Full Integration Challenges

- Vision Frontend (VFE)
  - Heterogeneous computation modules
    - Feature detection
    - Feature tracking
    - Stereo matching
    - Outliers rejection using RANSAC
    - …
- Backend (BE)
  - High dimensional and complex data structures
    - Large optimization problem (more than 4000 factors)
    - Dynamically changing factor graph
    - High computation precision (64-bit floating point)

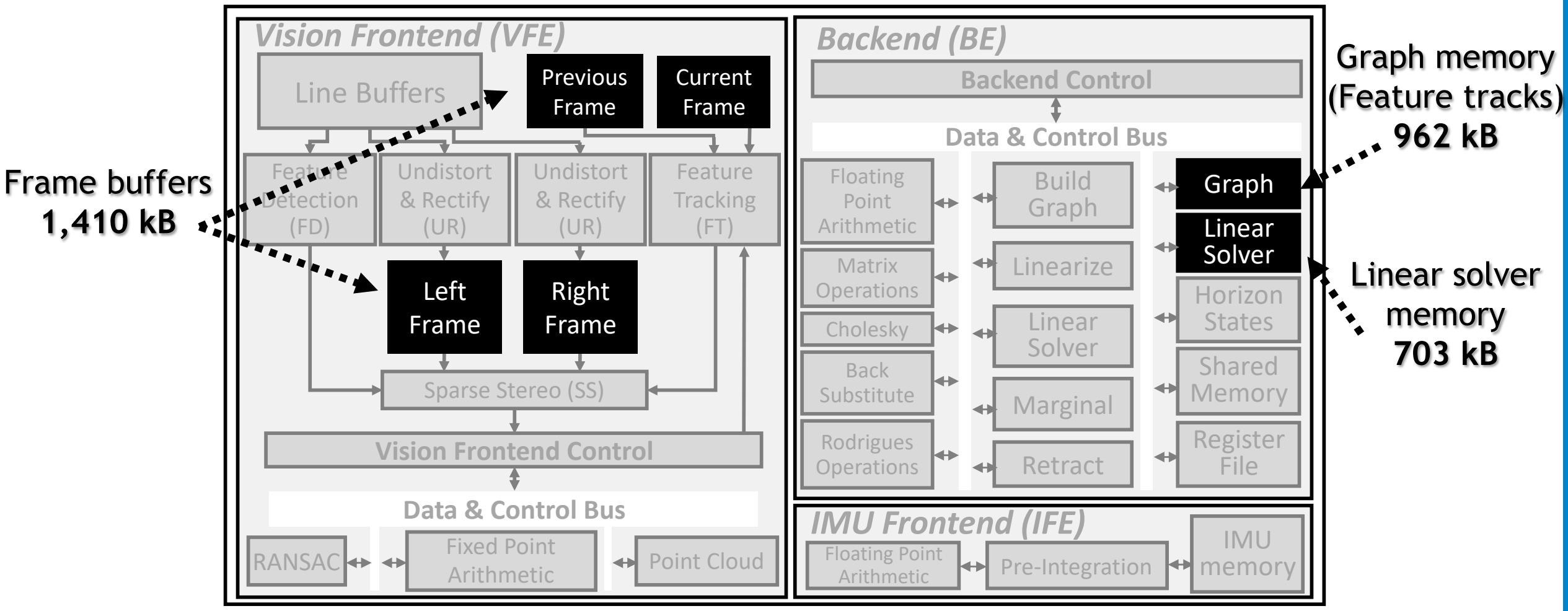
Outline

- Localization & Mapping: Visual-Inertial Odometry (VIO)
- Chip Architecture
- Main Contributions
- Chip Specifications and Comparisons
- Summary

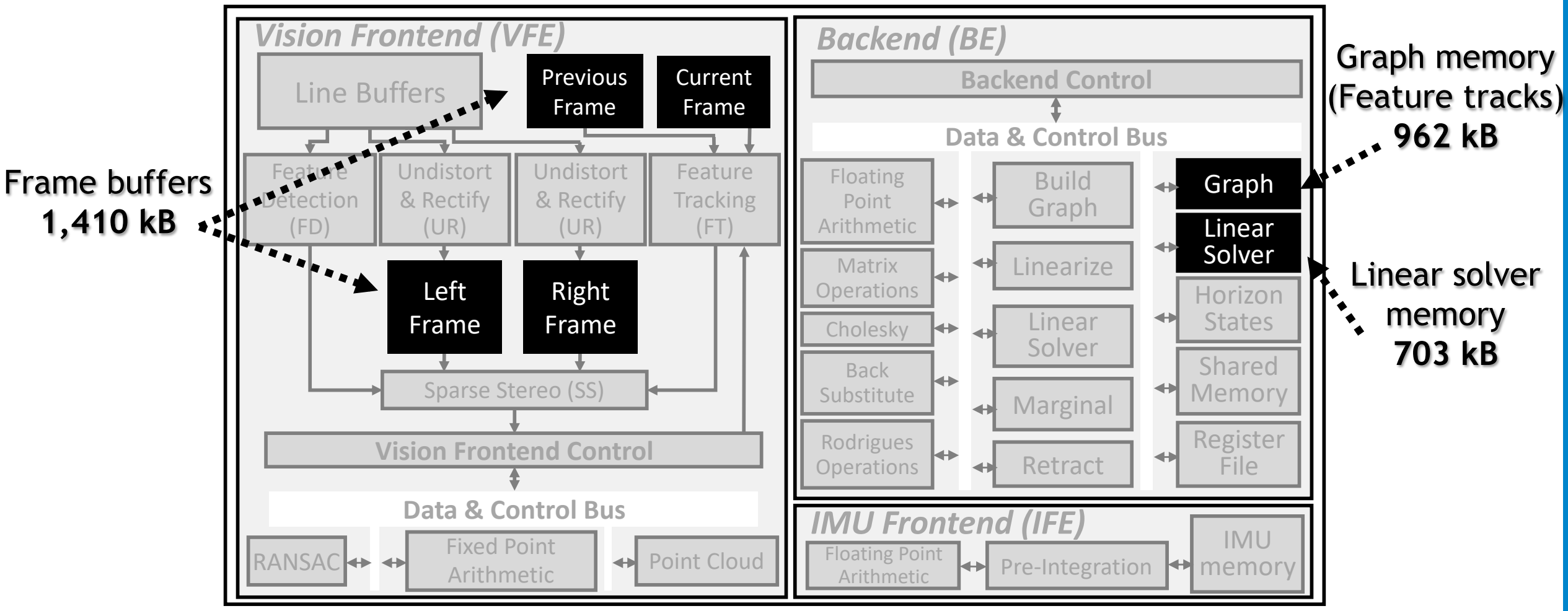
Enabling Full Integration



Enabling Full Integration



Enabling Full Integration



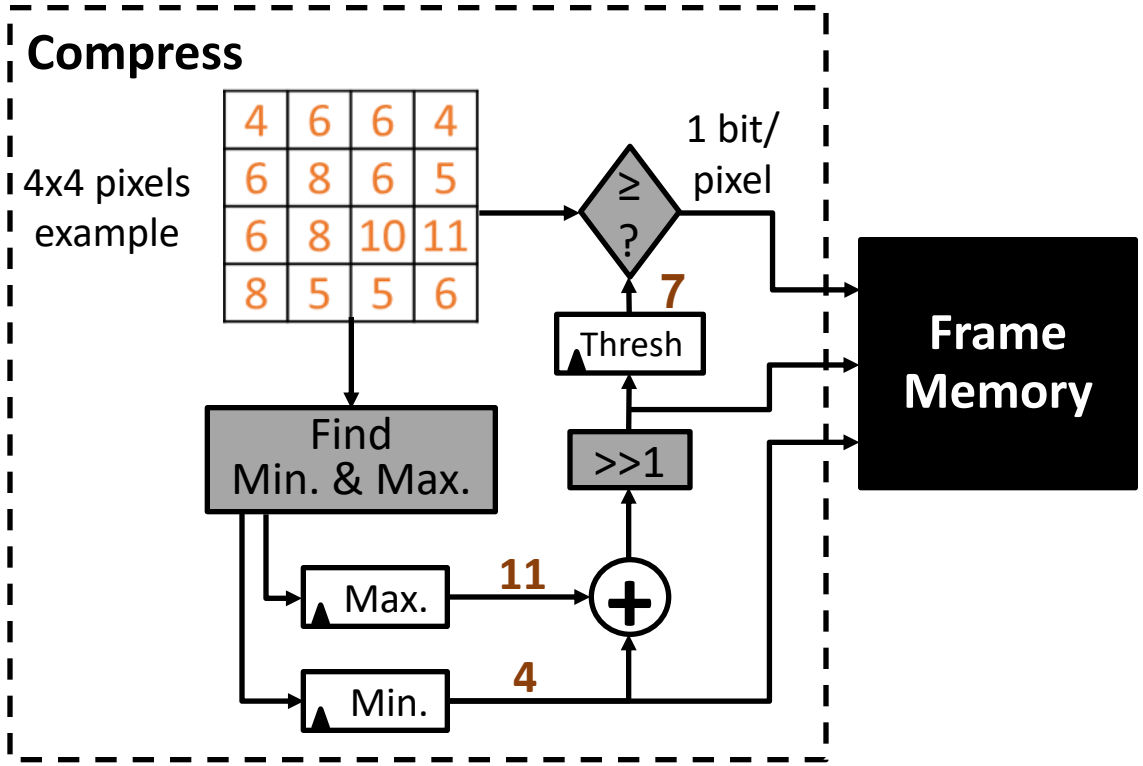
Use compression and exploit sparsity

Method 1

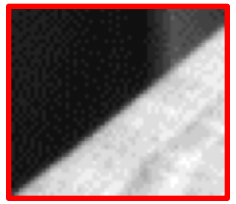
Data Compression

Frame Buffer: Image Compression

- Lossy Block-wise Image Compression



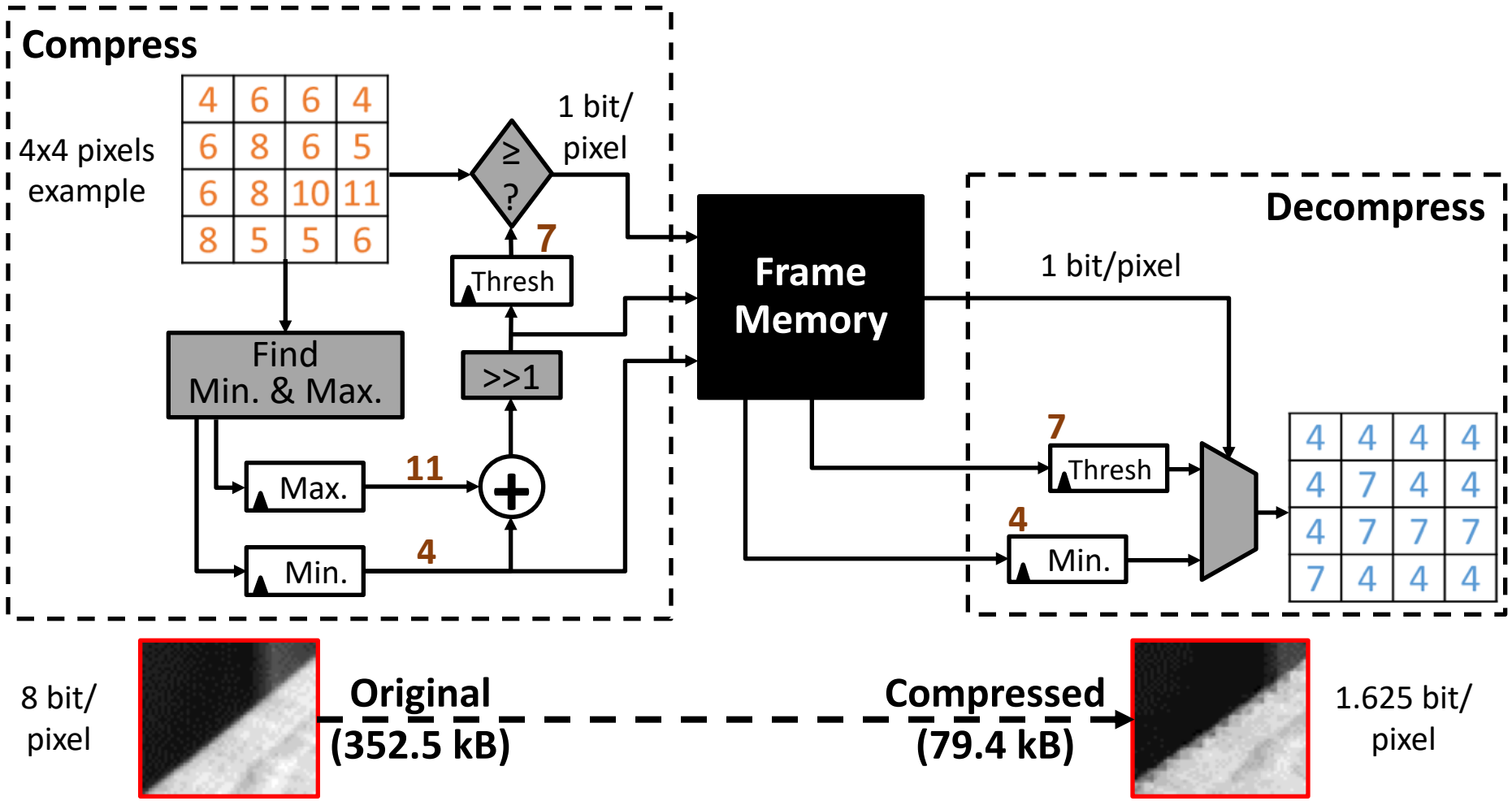
8 bit/ pixel



Original (352.5 kB)

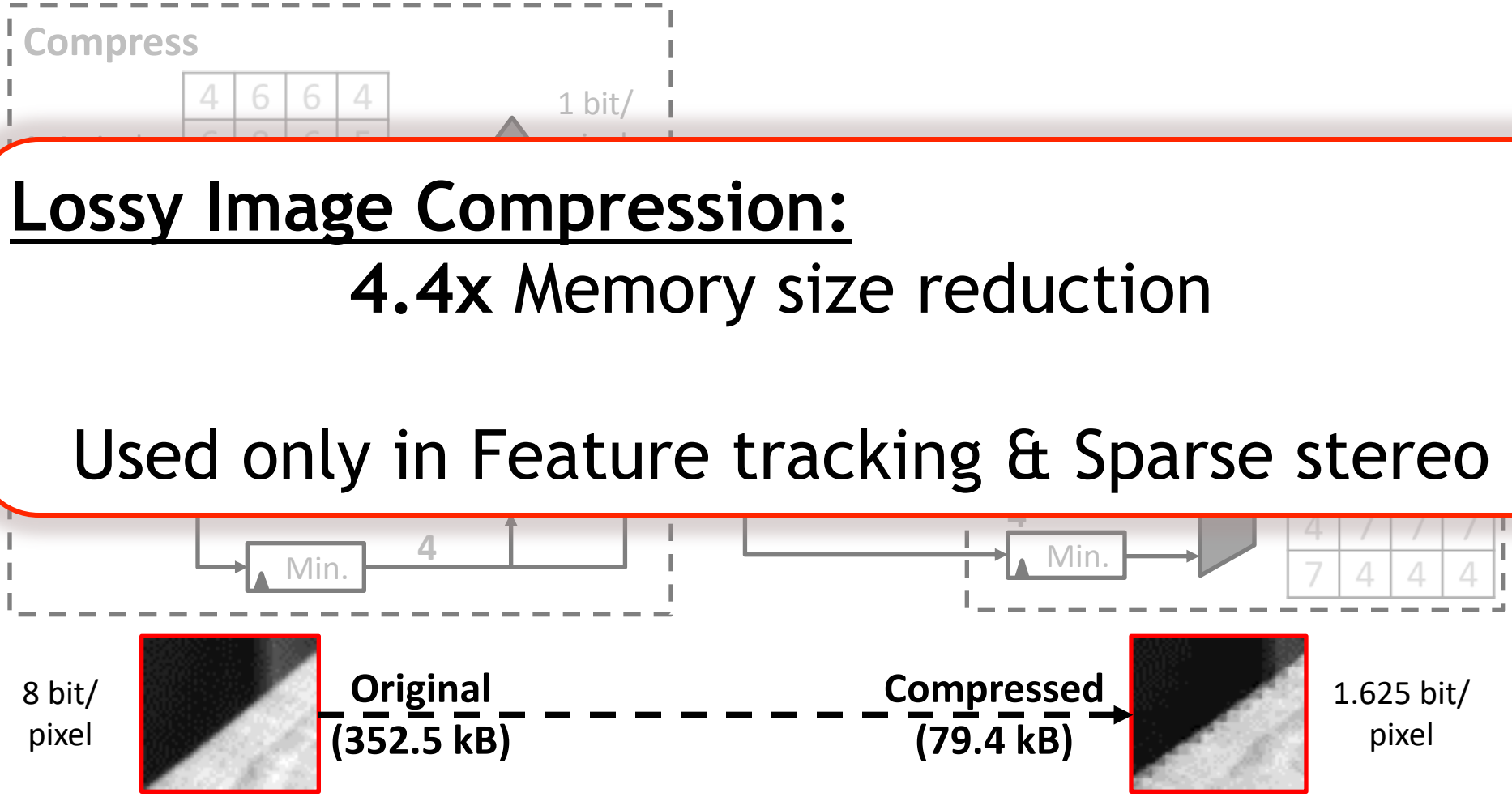
Frame Buffer: Image Compression

- Lossy Block-wise Image Compression



Frame Buffer: Image Compression

- Lossy Block-wise Image Compression



Lossy Image Compression:

4.4x Memory size reduction

Used only in Feature tracking & Sparse stereo

Method 2

Exploit Sparsity

(Structured & Unstructured)

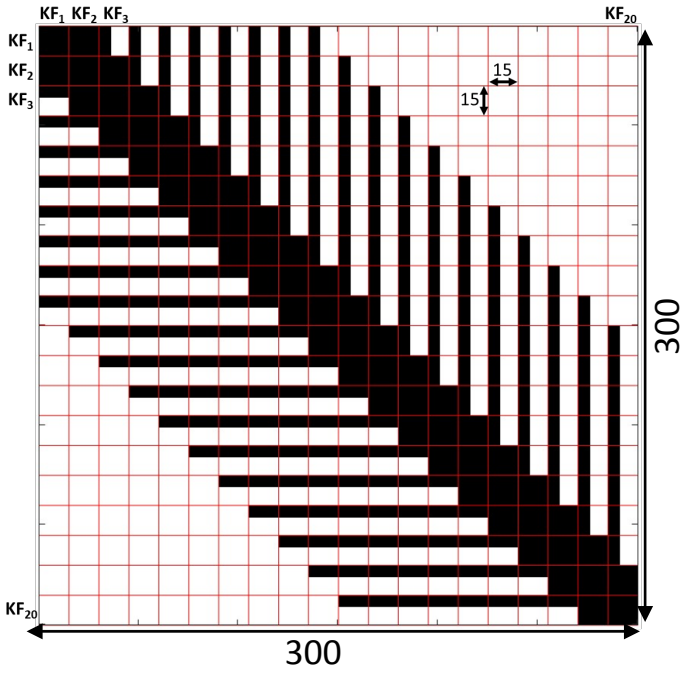
Linear Solver Memory: Structured Sparsity

$$\min_{x} \sum_{(i,j)\in\mathcal{F}} \left\| r_{\text{IMU}}\left(x, \Delta\tilde{R}_{ij}, \Delta\tilde{p}_{ij}, \Delta\tilde{v}_{ij}\right) \right\|^2 + \sum_{k\in\mathcal{L}} \sum_{i\in\mathcal{F}_k} \left\| r_{\text{CAM}}\left(x, l_k, u_{ik}^{l}, u_{ik}^{r}\right) \right\|^2 + \left\| r_{\text{PRIOR}}(x) \right\|^2$$

Linearize

$$\boldsymbol{H}\delta = \acute{\epsilon}$$

Solve a large linear system for δ



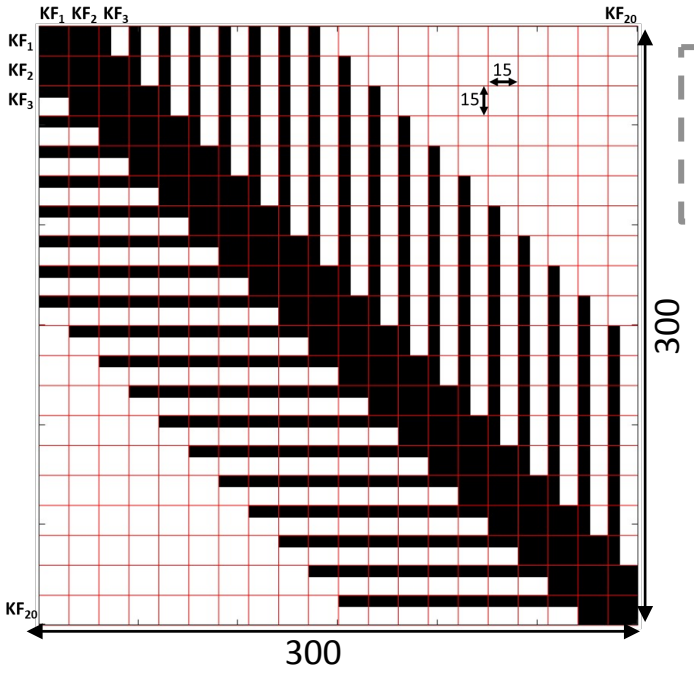
Linear Solver Memory: Structured Sparsity

$$\min_{x} \sum_{(i,j)\in\mathcal{F}} \left\| r_{\text{IMU}}\left(x, \Delta\tilde{R}_{ij}, \Delta\tilde{p}_{ij}, \Delta\tilde{v}_{ij}\right) \right\|^2 + \sum_{k\in\mathcal{L}} \sum_{i\in\mathcal{F}_k} \left\| r_{\text{CAM}}\left(x, l_k, u_{ik}^l, u_{ik}^r\right) \right\|^2 + \left\| r_{\text{PRIOR}}(x) \right\|^2$$

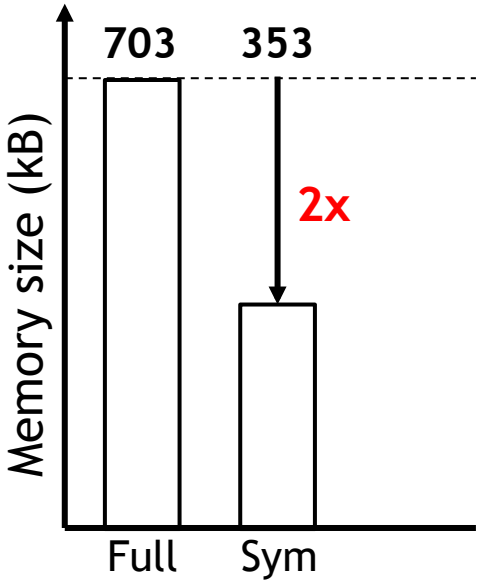
Linearize

$$H\delta = \acute{\epsilon}$$

Solve a large linear system for δ



H is:
- Symmetric



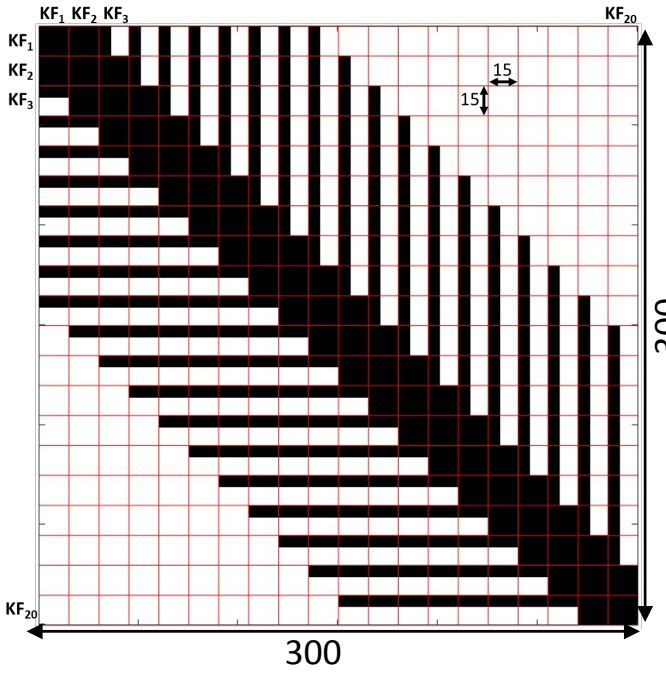
Linear Solver Memory: Structured Sparsity

$$\min_{x} \sum_{(i,j)\in\mathcal{F}} \left\| r_{\text{IMU}}\left(x, \Delta\tilde{R}_{ij}, \Delta\tilde{p}_{ij}, \Delta\tilde{v}_{ij}\right) \right\|^2 + \sum_{k\in\mathcal{L}} \sum_{i\in\mathcal{F}_k} \left\| r_{\text{CAM}}\left(x, l_k, u_{ik}^l, u_{ik}^r\right) \right\|^2 + \left\| r_{\text{PRIOR}}(x) \right\|^2$$

Linearize

$$H\delta = \acute{\epsilon}$$

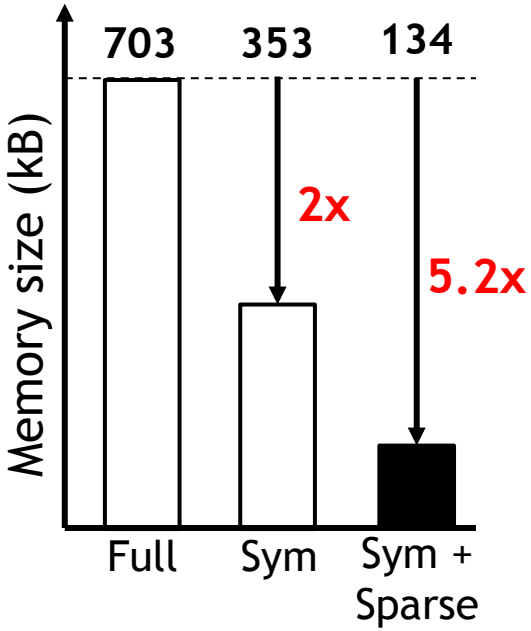
Solve a large linear system for δ



H is:

- Symmetric
- Sparse

(Black: non zero)



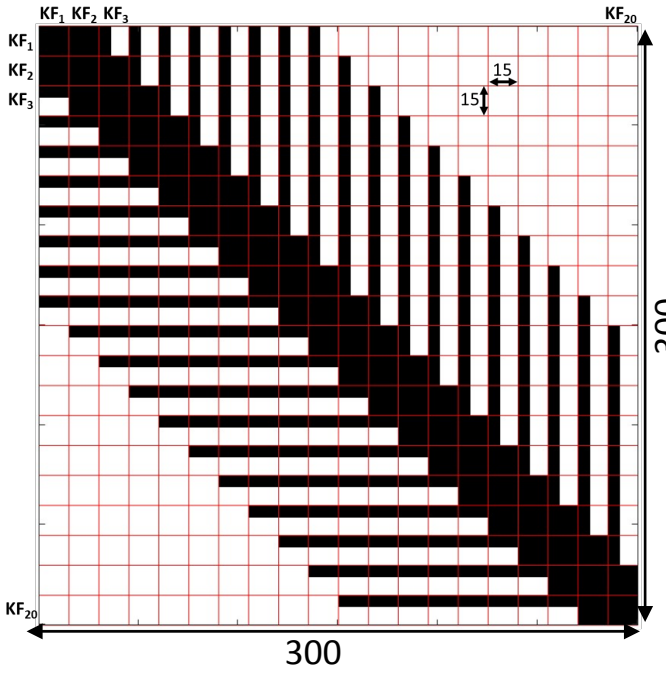
Linear Solver Memory: Structured Sparsity

$$\min_{x} \sum_{(i,j)\in\mathcal{F}} \left\| r_{\text{IMU}}\left(x, \Delta\tilde{R}_{ij}, \Delta\tilde{p}_{ij}, \Delta\tilde{v}_{ij}\right)\right\|^2 + \sum_{k\in\mathcal{L}} \sum_{i\in\mathcal{F}_k} \left\| r_{\text{CAM}}\left(x, l_k, u_{ik}^l, u_{ik}^r\right)\right\|^2 + \left\| r_{\text{PRIOR}}(x)\right\|^2$$

Linearize

$$H\delta = \acute{\epsilon}$$

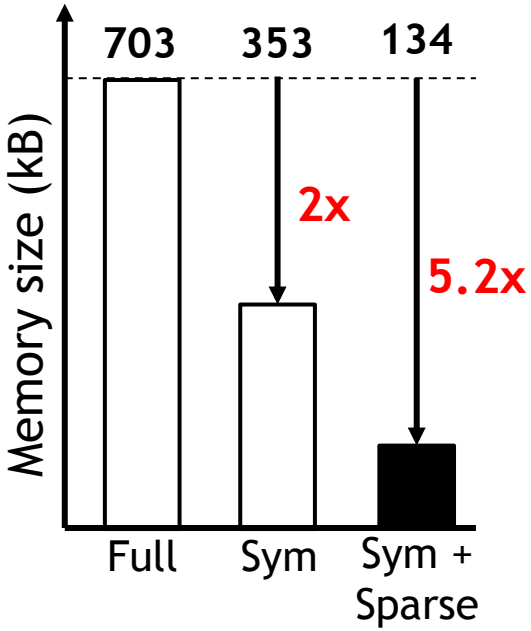
Solve a large linear system for δ

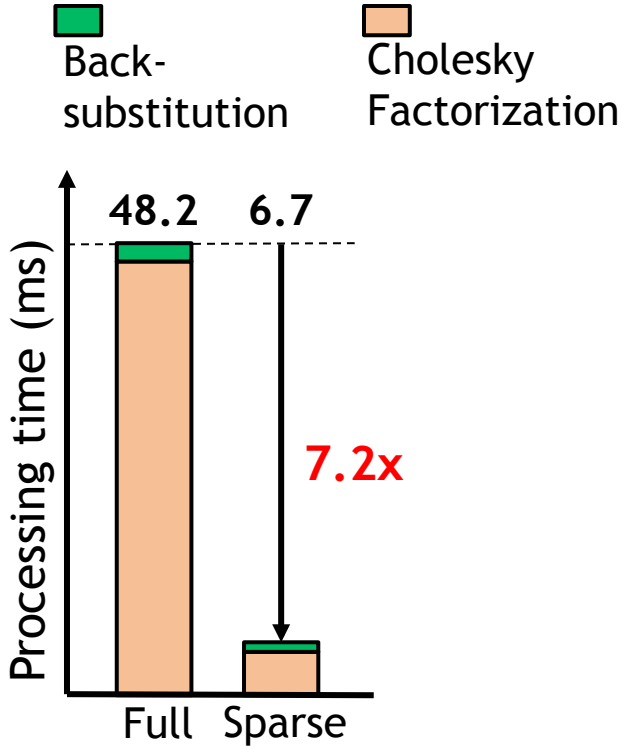


H is:

- Symmetric
- Sparse

(Black: non zero)





Linear Solver Memory: Structured Sparsity

$$\min_x \sum_{(i,j)\in\mathcal{F}} \left\| r_{\text{IMU}}\left(x, \Delta\tilde{R}_{ij}, \Delta\tilde{p}_{ij}, \Delta\tilde{v}_{ij}\right)\right\|^2 + \sum_{k\in\mathcal{L}}\sum_{i\in\mathcal{F}_k} \left\| r_{\text{CAM}}\left(x, l_k, u_{ik}^l, u_{ik}^r\right)\right\|^2 + \left\| r_{\text{PRIOR}}(x)\right\|^2$$

Linearize

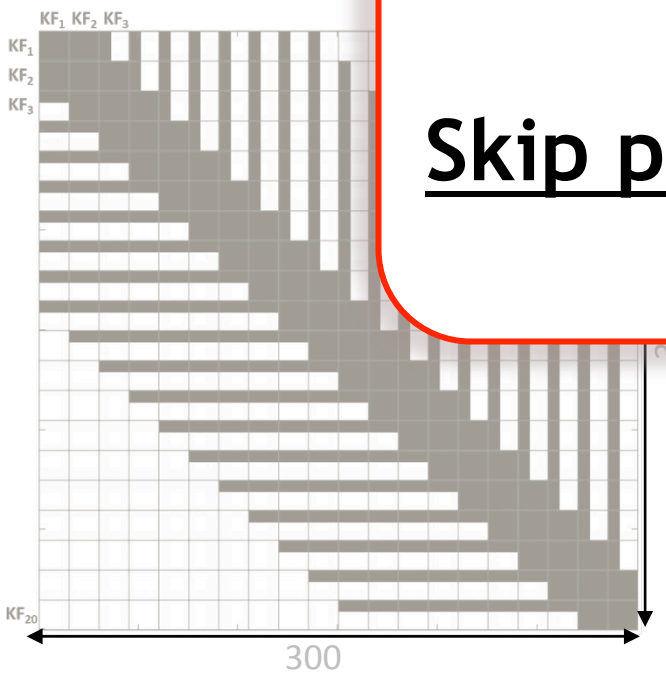
$$H\delta = \acute{\varepsilon}$$

Storing symmetric non-zero values:

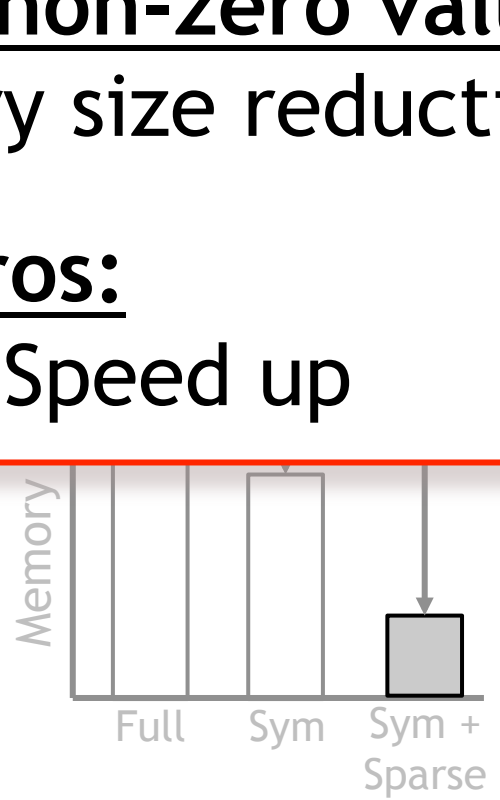
5.2x Memory size reduction

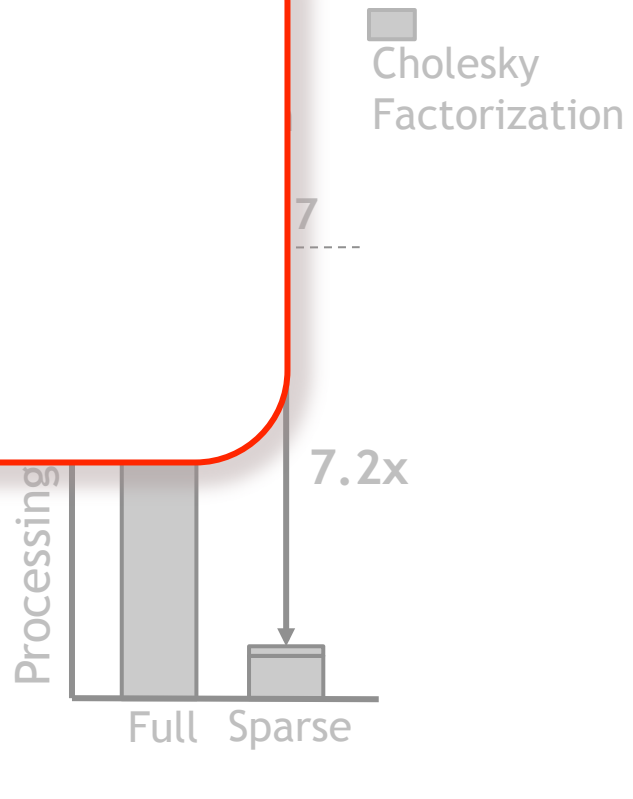
Skip processing zeros:

7.2x Speed up



(Black: non zero)

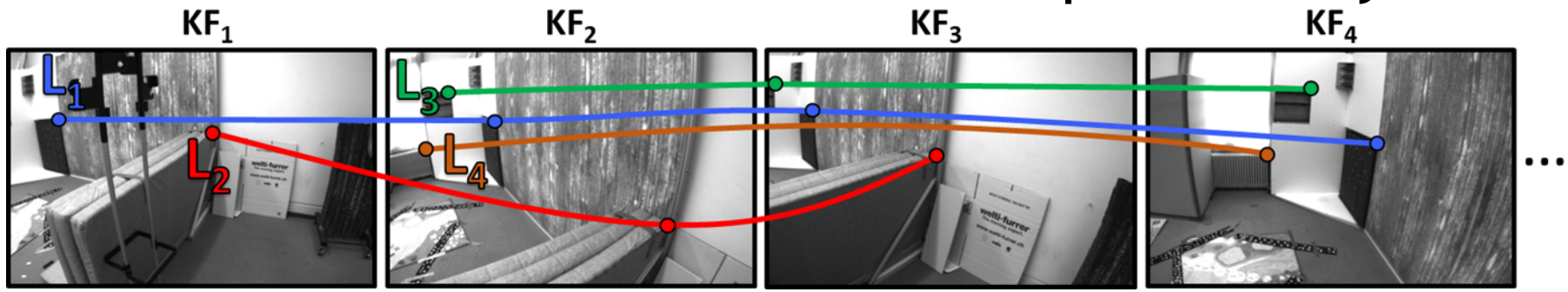




Cholesky Factorization

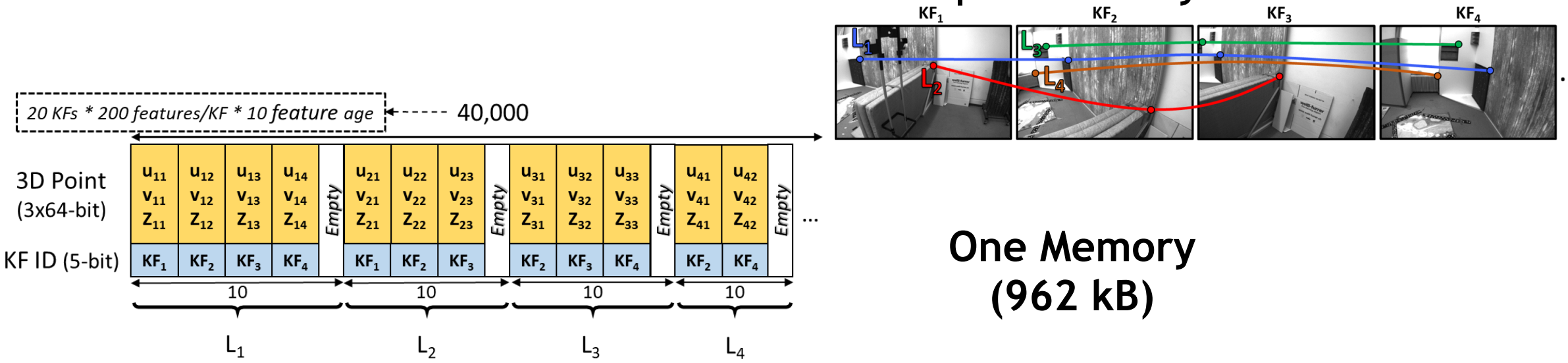
Feature Tracks: Unstructured Sparsity

- Feature Tracks accounts for 88% of the Graph memory



Feature Tracks: Unstructured Sparsity

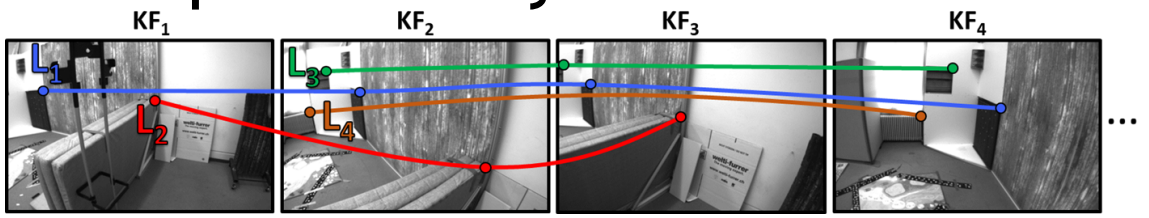
- Feature Tracks accounts for 88% of the Graph memory

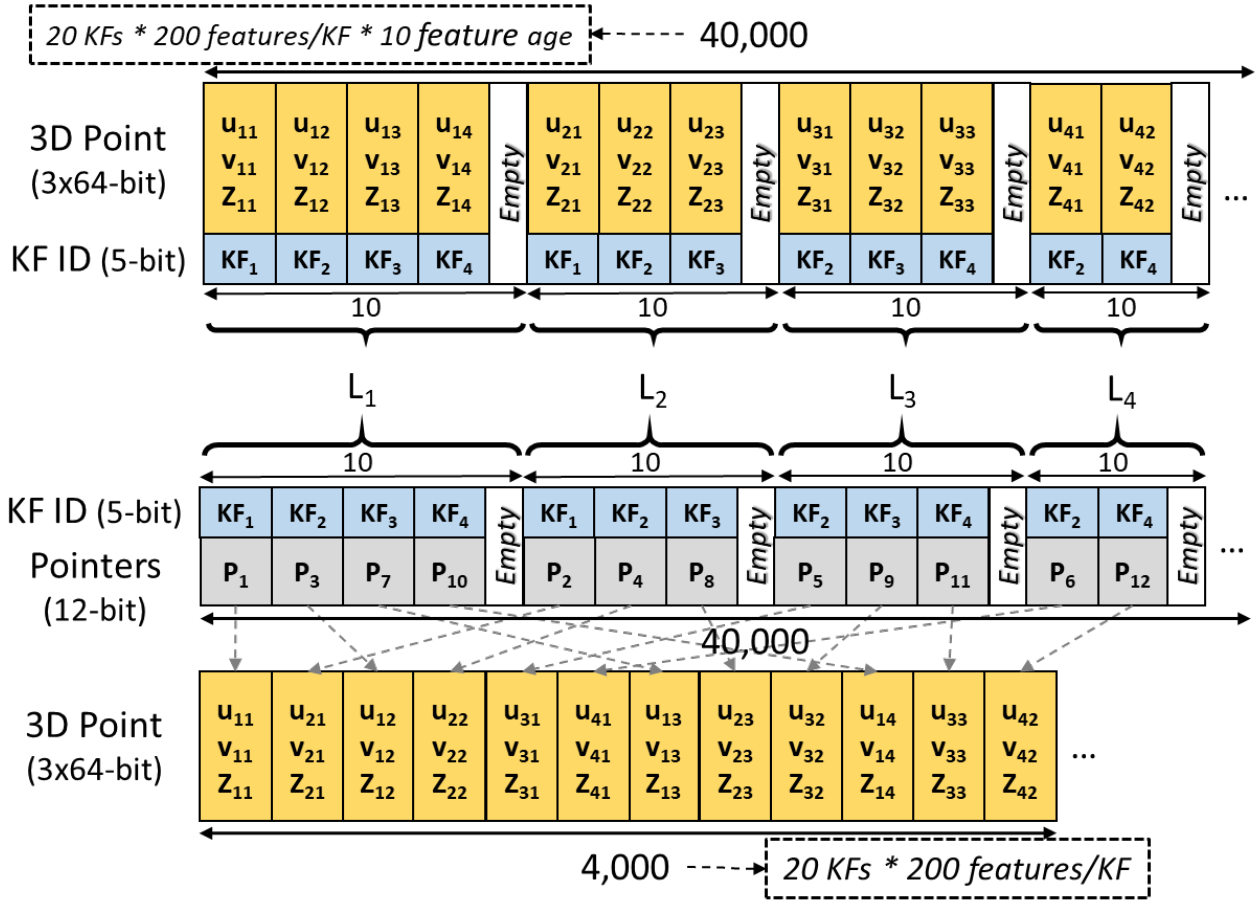


One Memory (962 kB)

Feature Tracks: Unstructured Sparsity

- Feature Tracks accounts for 88% of the Graph memory





One Memory (962 kB)

Two-stage Memory (177 kB)

Feature Tracks: Unstructured Sparsity

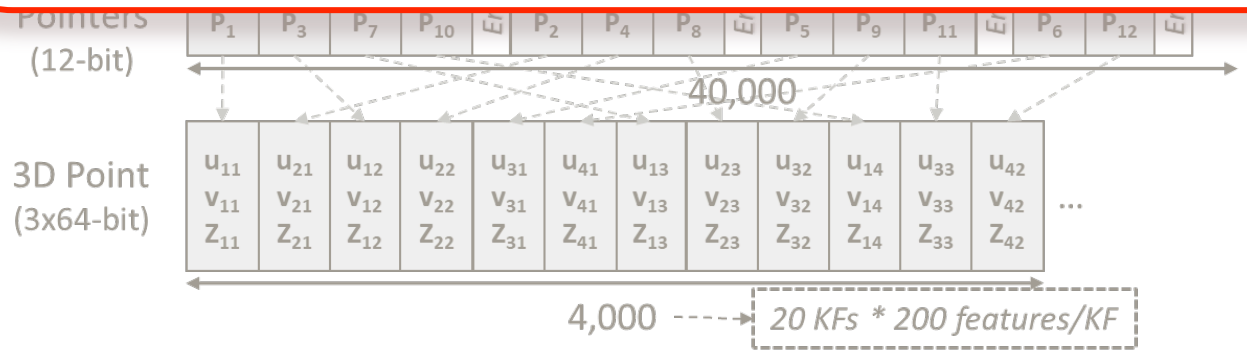
- Feature Tracks accounts for 88% of the Graph memory

Feature tracks two-stage storage:

5.4x Memory size reduction

Overhead:

1 extra cycle access latency



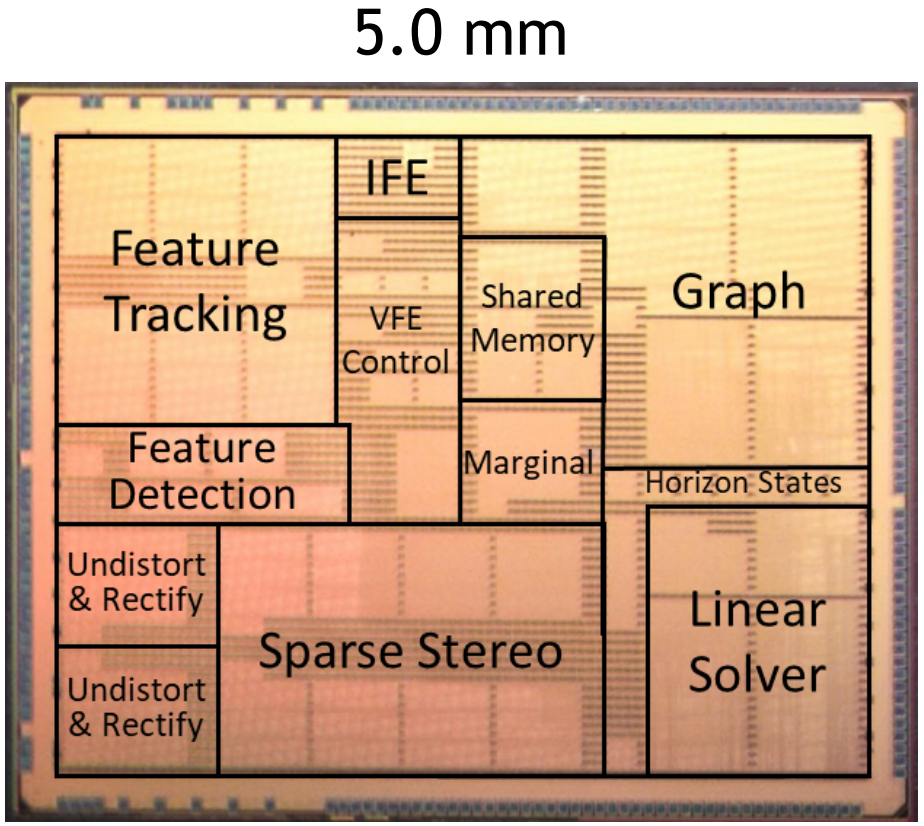
One Memory (962 kB)

Two-stage Memory (177 kB)

Outline

- Localization & Mapping: Visual-Inertial Odometry (VIO)
- Chip Architecture
- Main Contributions
- Chip Specifications and Comparisons
- Summary

Navion Chip

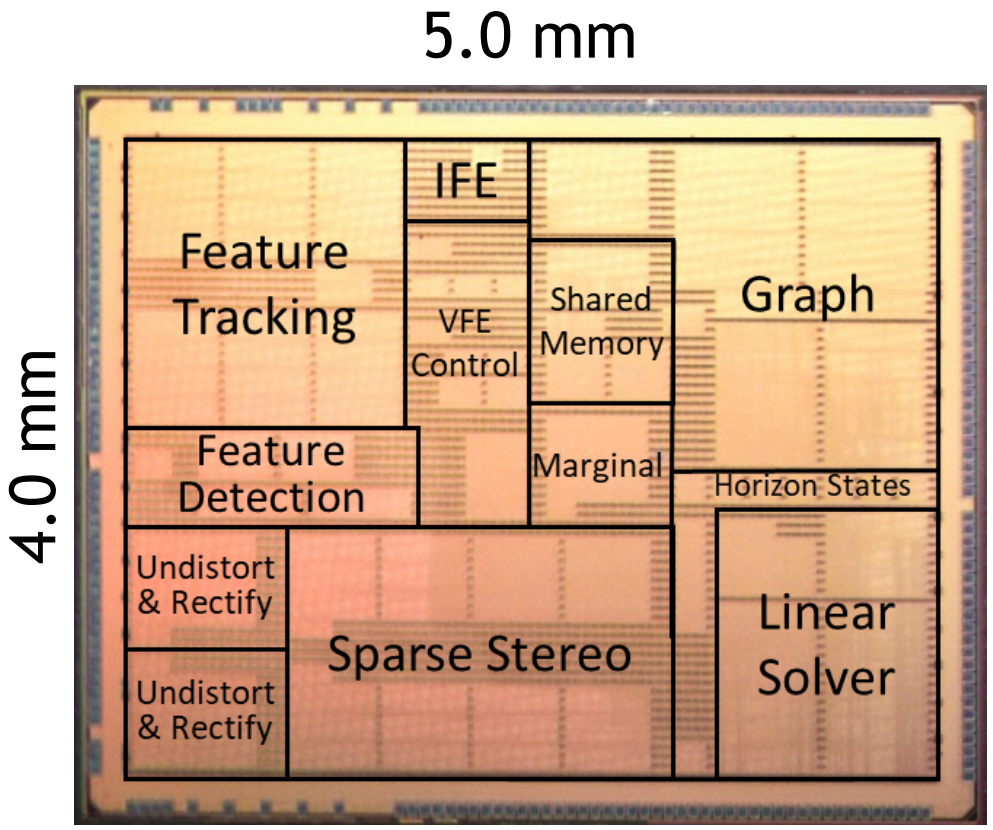


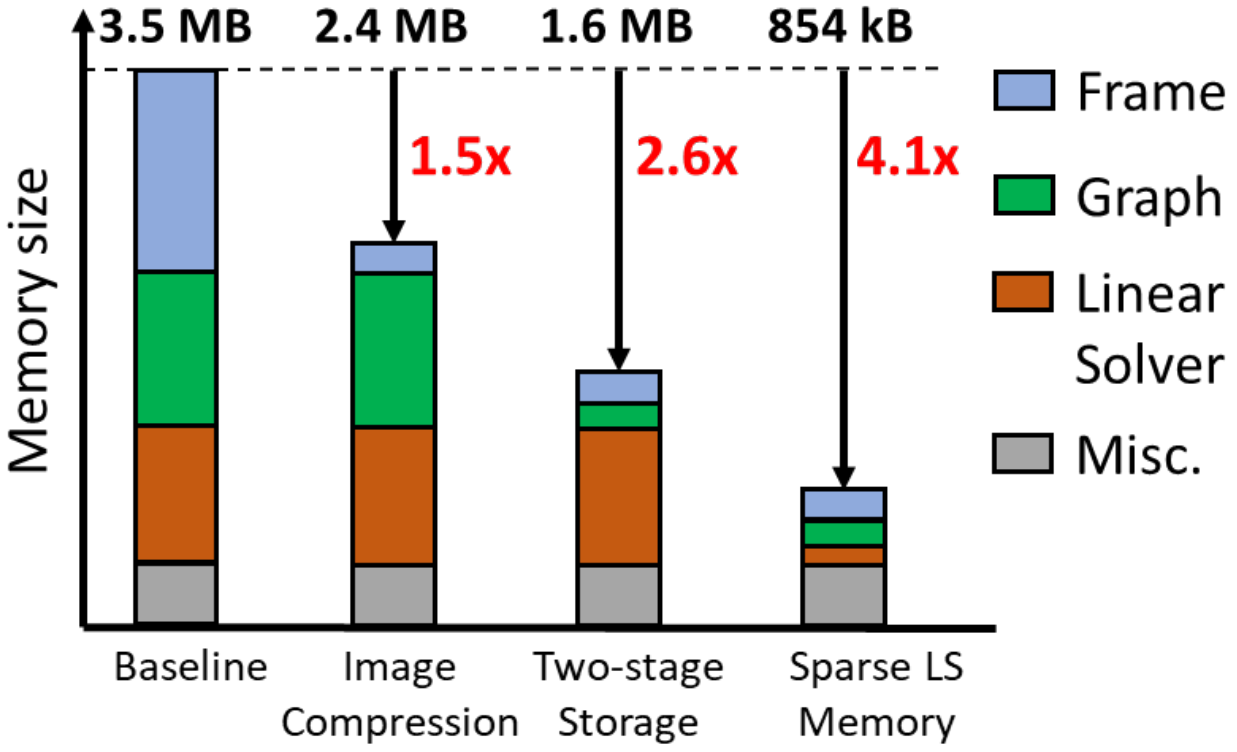
Silicon Specifications

Technology	65nm CMOS
Chip area (mm^2)	4.0 x 5.0
Logic gates	2,043 kgates
Resolution	752 x 480
SRAM	854 kB

Over 250 configurable parameters to adapt to different sensors and environments

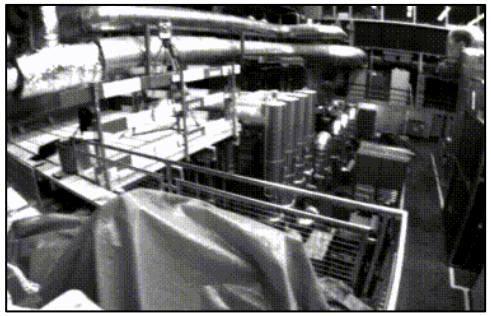
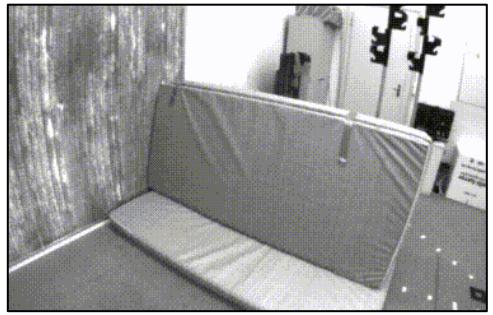
Memory Optimization



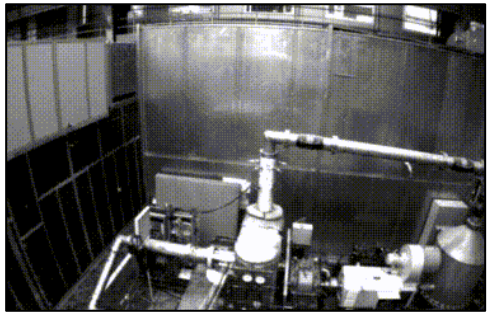


Navion Evaluation

- EuRoC dataset
  - A very challenging, and widely used UAV dataset
  - 11 sequences with three categories: easy, medium & difficult

Examples of Easy Sequences




Dark scenes *Motion blur*

Examples of Difficult Sequences

Navion Evaluation

- **Peak Performance @ Max Configuration**
  - VFE: 28 – 171 fps (71 fps average)
  - BE: 16 – 90 fps (19 fps average)
- Average Power Consumption: 24mW
- Trajectory Error: 0.28%

- **Real-Time Performance @ Optimized Configuration**
  - VF: 20 fps
  - BE: 5 fps
- Average Power Consumption: 2mW
- Trajectory Error: 0.27%

Navion Evaluation

- Average numbers over the 11 EuRoC dataset sequences

Platform	**Xeon** (E5-2667)	**ARM** (Cortex A15)	**Navion** (Peak w/ Max Config)	**Navion** (Real-time w/ Optimized Config)
Trajectory Error (%)	0.22%		0.28%	0.27%
Camera rate (fps)	63	19	**71**	**20**
Keyframe rate (fps)	12	2	**19**	**5**
Average Power (W)	27.9	2.4	**0.024**	**0.002**
Energy (mJ/KF)	3,638	1,573	**2.3**	**0.7**

Navion Evaluation

- Average numbers over the 11 EuRoC dataset sequences

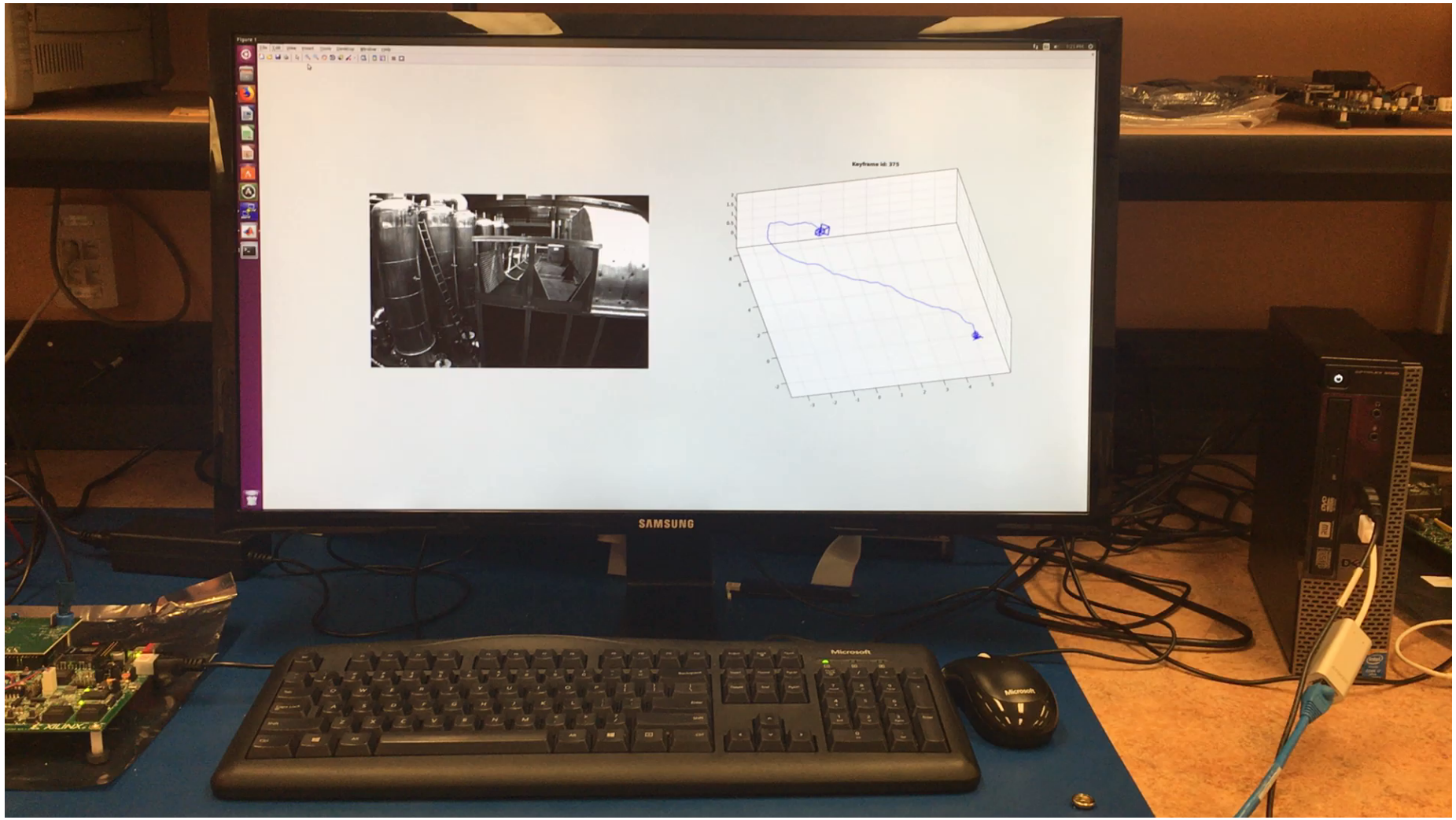
Platform	Xeon (E5-2667)	ARM (Cortex A15)	Navion (Peak w/ Max Config)	Navion (Real-time w/ Optimized Config)
Trajectory Error (%)	0.22%		0.28%	0.27%
Camera rate (fps)	63	19	**71**	**20**
Keyframe rate (fps)	12	2	**19**	**5**
Average Power (W)	27.9	2.4	**0.024**	**0.002**
Energy (mJ/KF)	3,638	1,573	**2.3**	**0.7**

Navion Energy:

684x or 2,247x less than embedded ARM CPU

1,582x or 5,197x less than server Xeon CPU

Navion System Demo



https://youtu.be/X5VZkPo_704

Outline

- Localization & Mapping: Visual-Inertial Odometry (VIO)
- Chip Architecture
- Main Contributions
- Chip Specifications and Comparisons
- Summary

Summary

- First **full integration** of VIO pipeline on chip for robot perception

Summary

- First **full integration** of VIO pipeline on chip for robot perception
- Leverage compression and sparsity to reduce memory size
  - **4.4x** reduction with image compression
  - **5.2x** reduction with structured sparsity in linear solver
  - **5.4x** reduction with unstructured sparsity in feature tracks

Summary

- First **full integration** of VIO pipeline on chip for robot perception
- Leverage compression and sparsity to reduce memory size
  - **4.4x** reduction with image compression
  - **5.2x** reduction with structured sparsity in linear solver
  - **5.4x** reduction with unstructured sparsity in feature tracks
- Navion is **2 to 3 orders of magnitude** more energy efficient than CPU

Summary

- First **full integration** of VIO pipeline on chip for robot perception
- Leverage compression and sparsity to reduce memory size
  - **4.4x** reduction with image compression
  - **5.2x** reduction with structured sparsity in linear solver
  - **5.4x** reduction with unstructured sparsity in feature tracks
- Navion is **2 to 3 orders of magnitude** more energy efficient than CPU

Acknowledgment

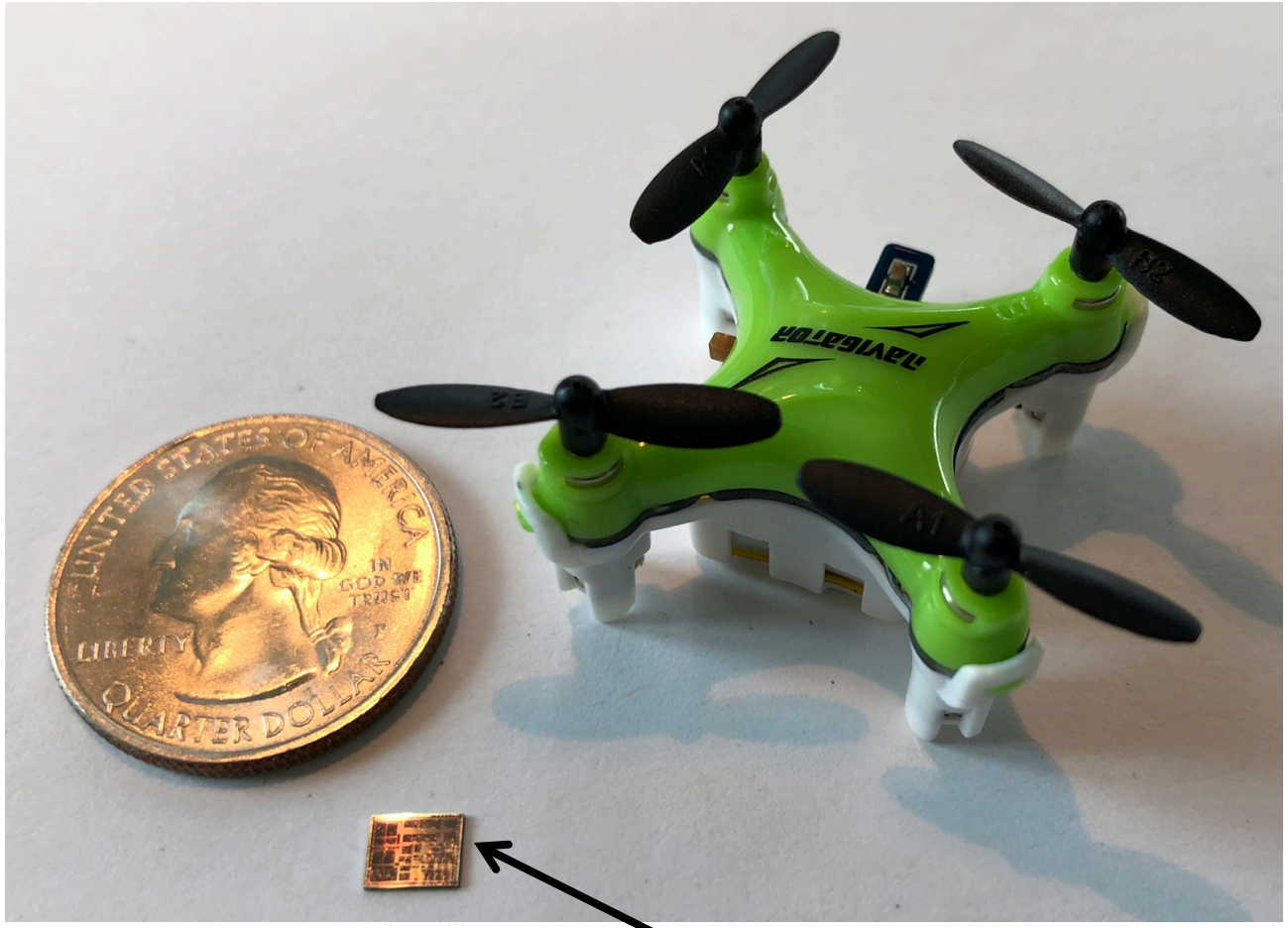
AFOSR YIP and NSF CAREER

References

http://navion.mit.edu

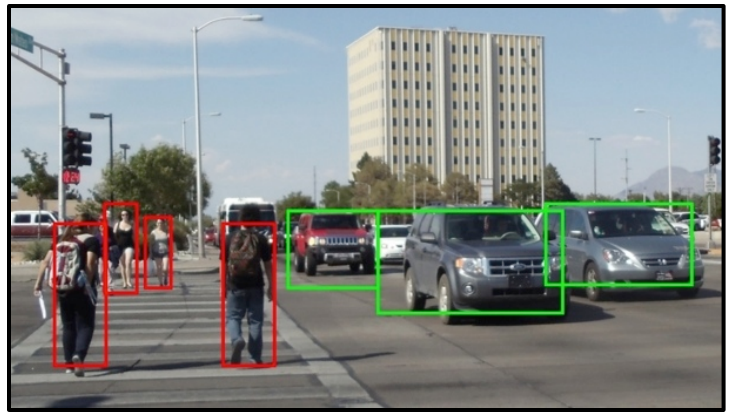
- A. Suleiman, Z. Zhang, L. Carlone, S. Karaman, V. Sze, "**Navion: A Fully Integrated Energy-Efficient Visual-Inertial Odometry Accelerator for Autonomous Navigation of Nano Drones**," *IEEE Symposium on VLSI Circuits (VLSI-Circuits)*, June 2018.

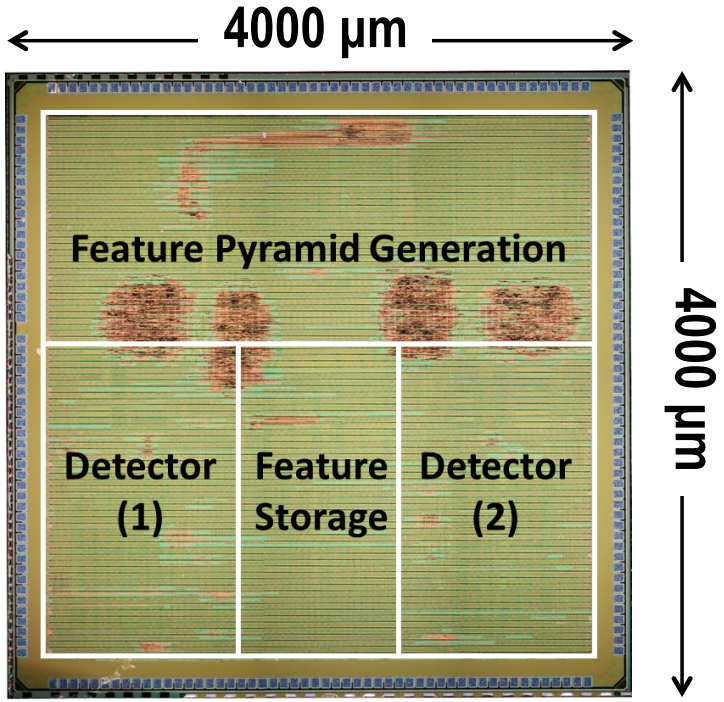
- Z. Zhang*, A. Suleiman*, L. Carlone, V. Sze, S. Karaman, "**Visual-Inertial Odometry on Chip: An Algorithm-and-Hardware Co-design Approach**," *Robotics: Science and Systems (RSS)*, July 2017.



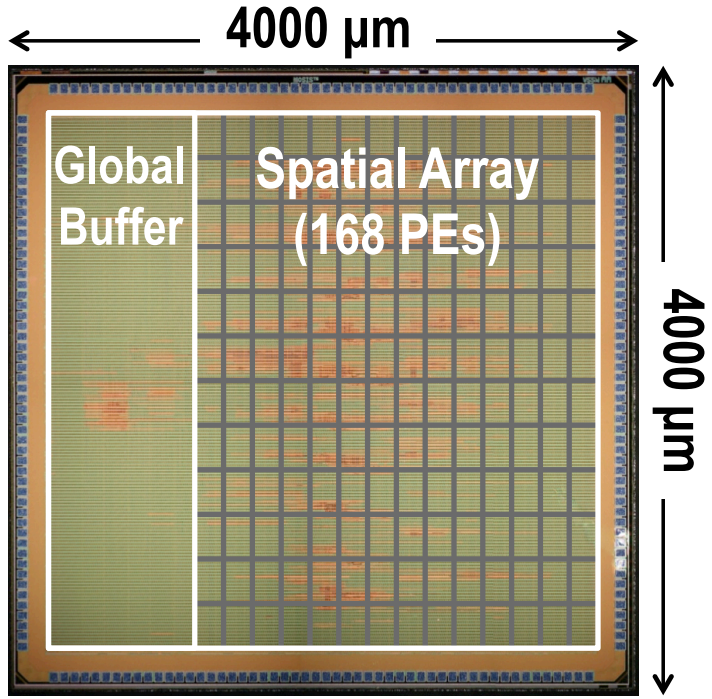
Other Related Works

Semantic Understanding





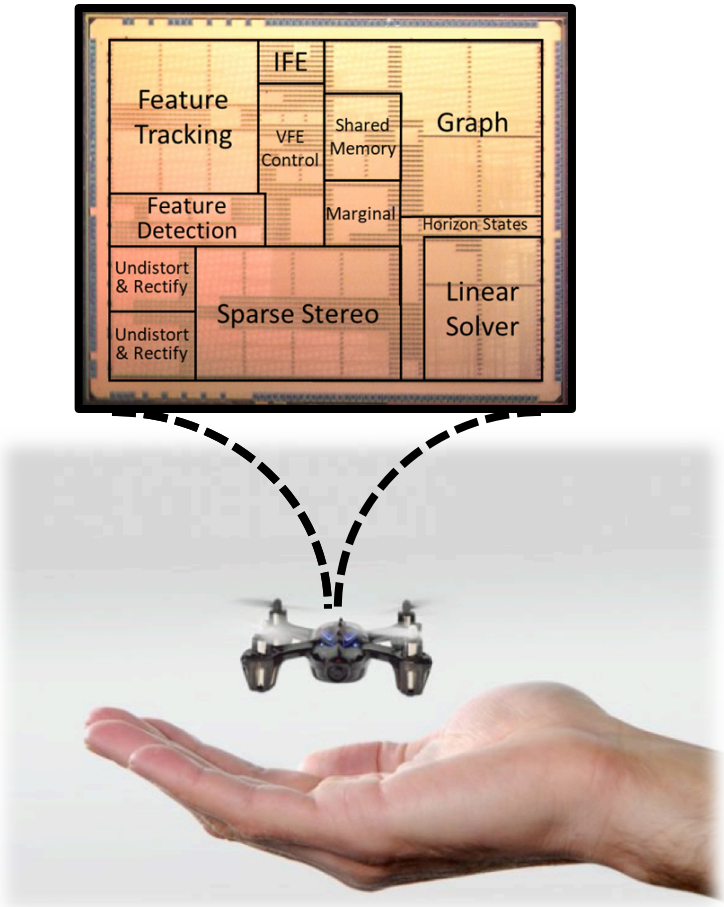
Object Detection
w/ Deformable Parts Models
[Suleiman et al., VLSI 2016]



Eyeriss: **Deep Neural Networks**
[ISSCC 2016, ISCA 2016]
Eyeriss v2
https://arxiv.org/abs/1807.07928

Website: http://www.rle.mit.edu/eems/

Questions



http://navion.mit.edu/