Real Time Identifiction in Crowds

J L Gouws[1]

Supervisor: Mr. J Connan[1]

[1]Rhodes University

28 March 2022

Predator[1]

Predator [1]

Why the TLD approach?

Why the TLD approach?

- Fast and Lightweight
-

Why the TLD approach?

- Fast and Lightweight
- No prior “training”
-

Why the TLD approach?

- Fast and Lightweight
- No prior “training”
- Learns quickly
-

Why the TLD approach?

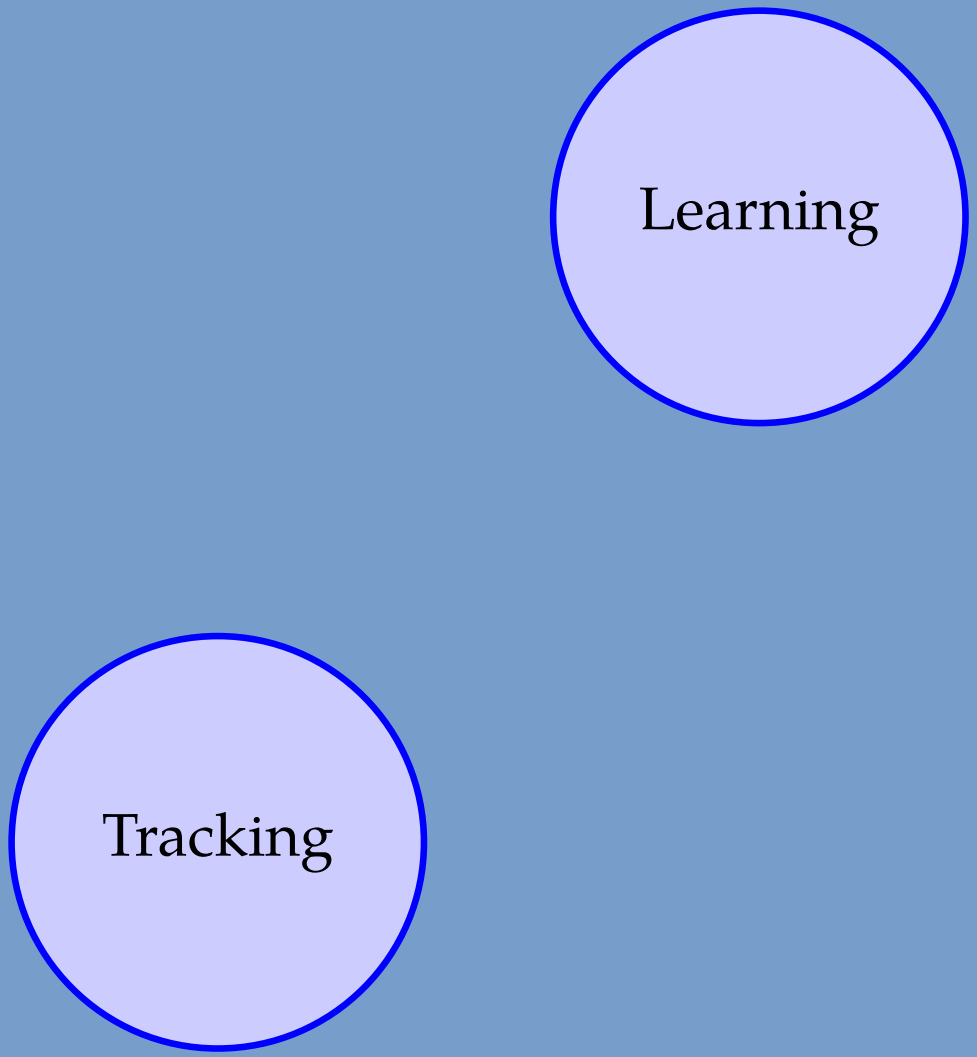
- Fast and Lightweight
- No prior “training”
- Learns quickly
- Synergy with recognition

Why the TLD approach?

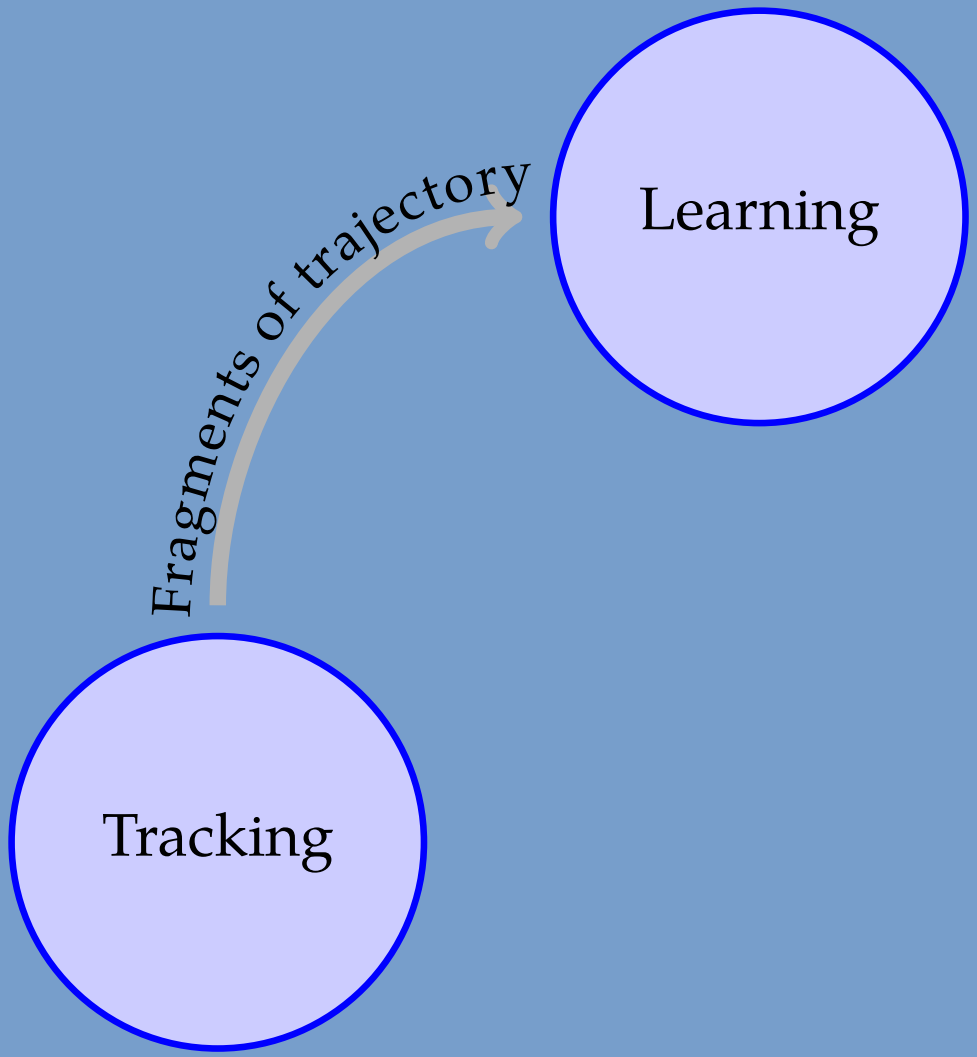
- Fast and Lightweight
- No prior “training”
- Learns quickly
- Synergy with recognition
- Generic

TLD Approach[2]

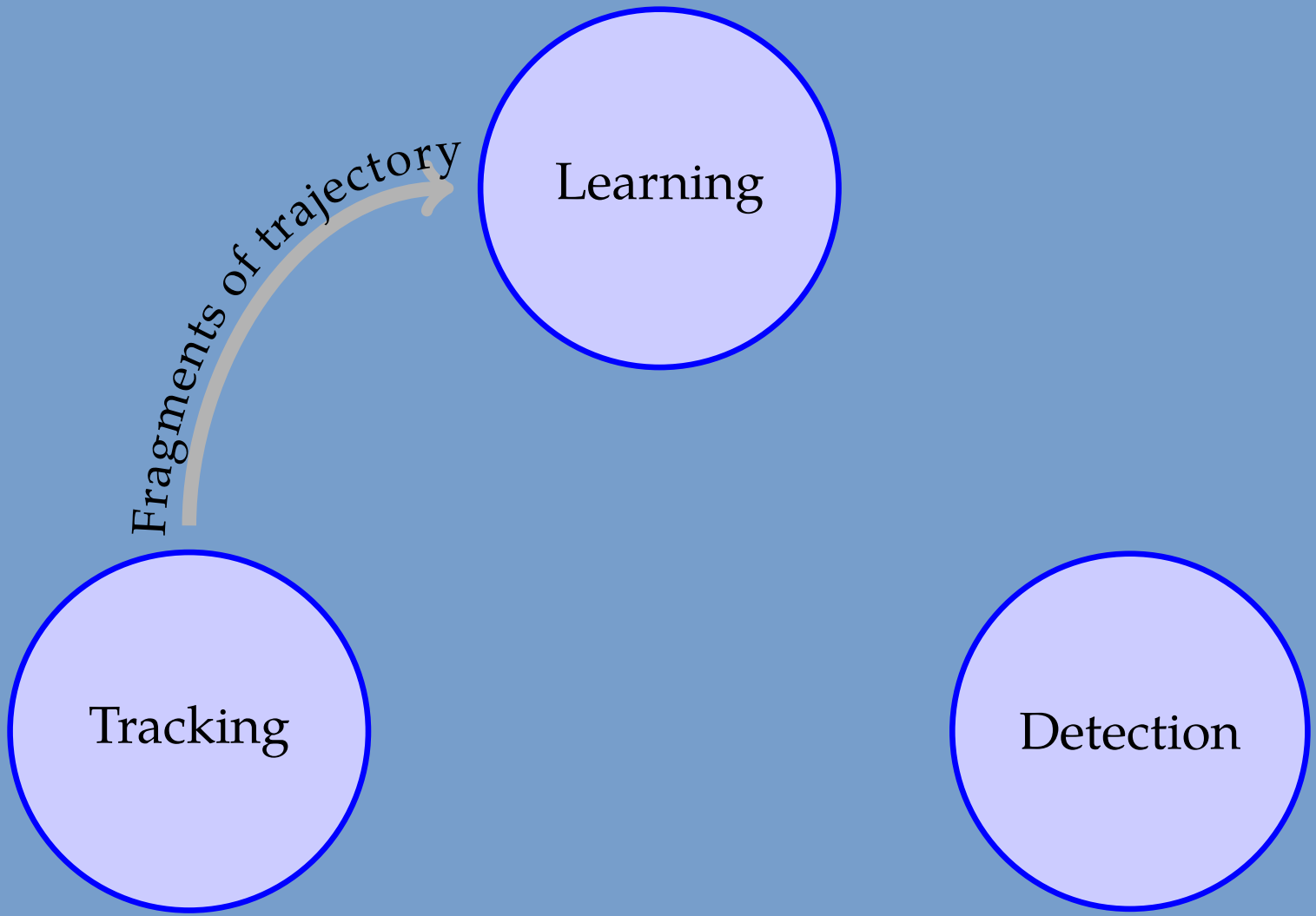
TLD Approach[2]



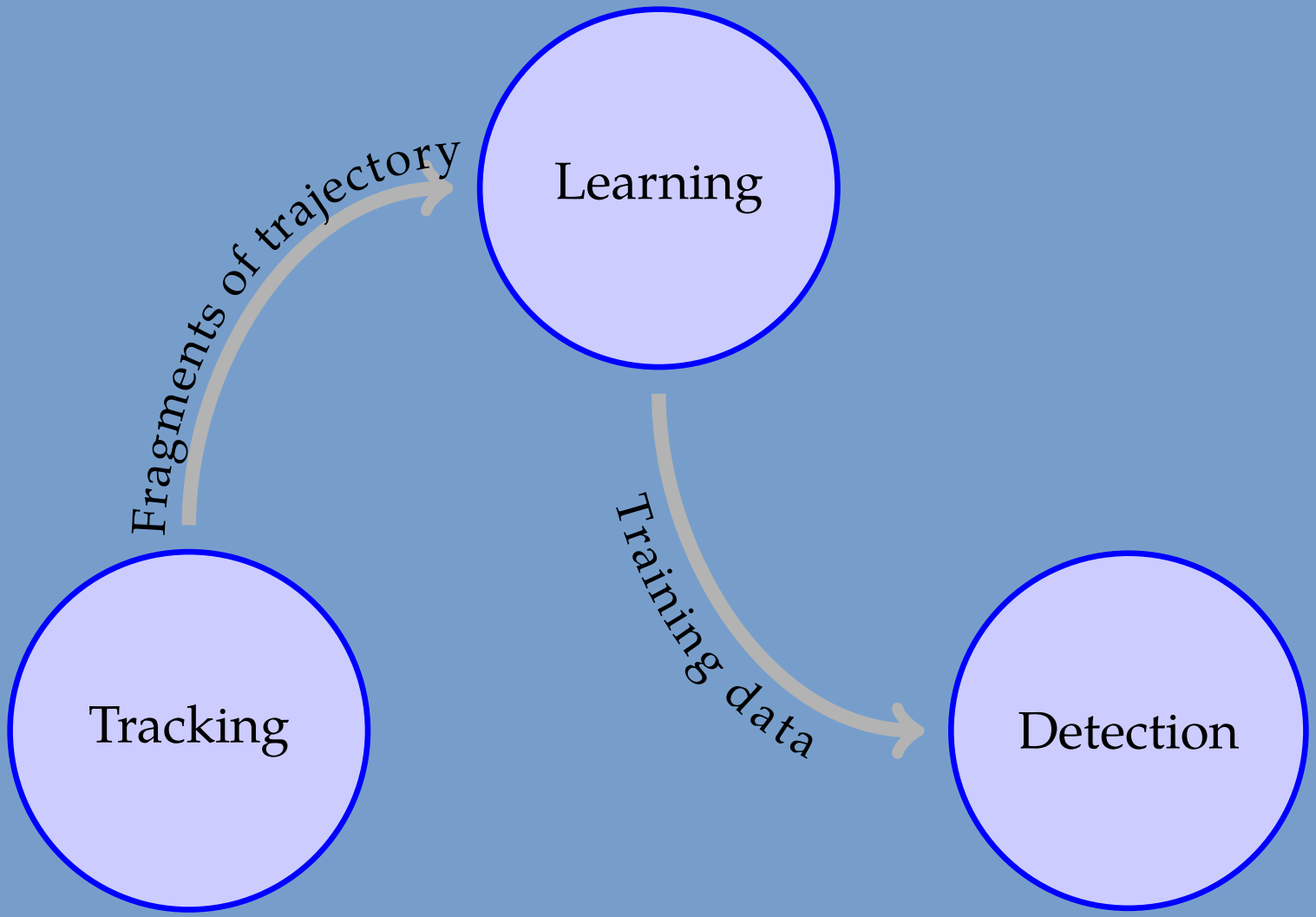
TLD Approach[2]



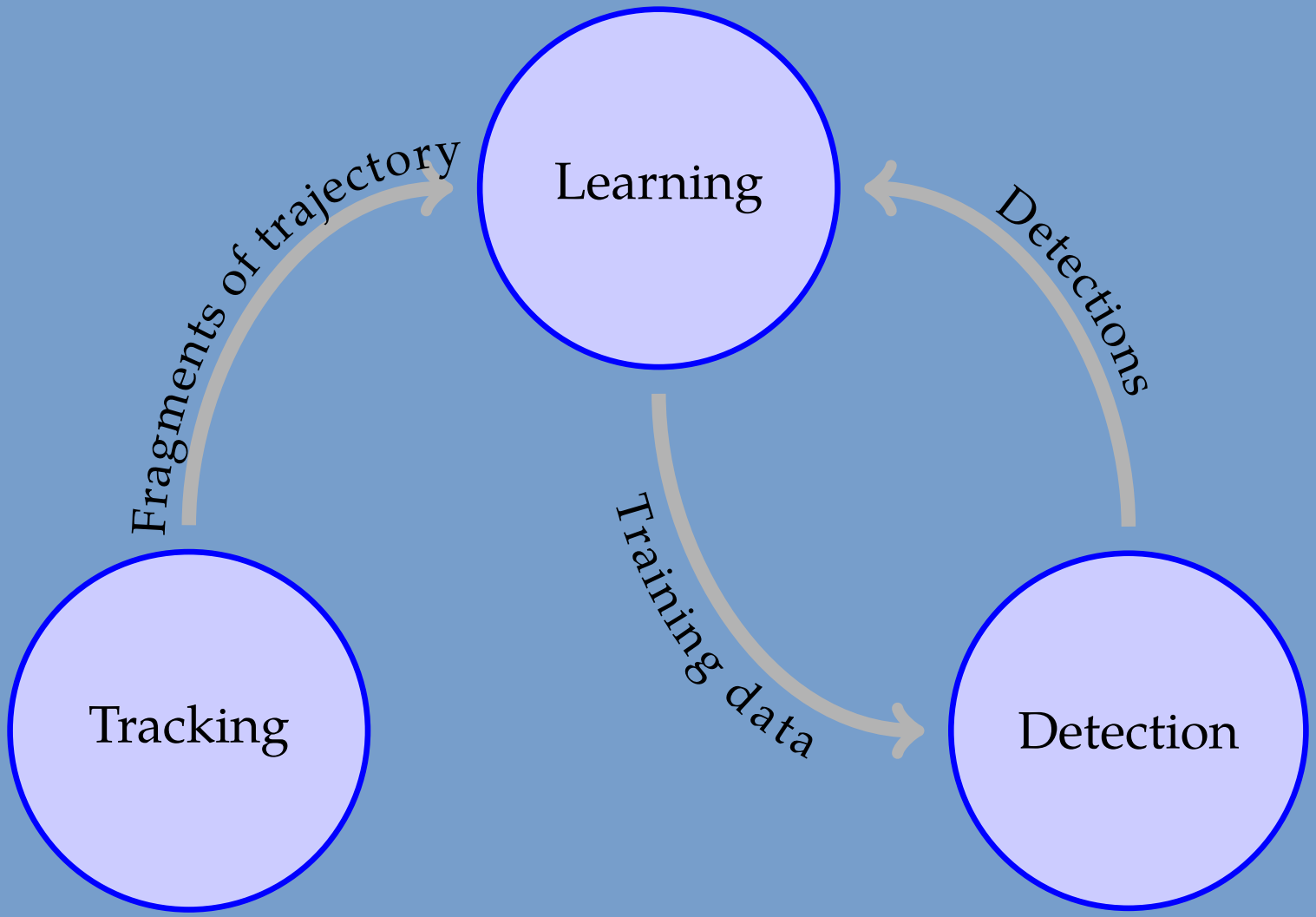
TLD Approach[2]



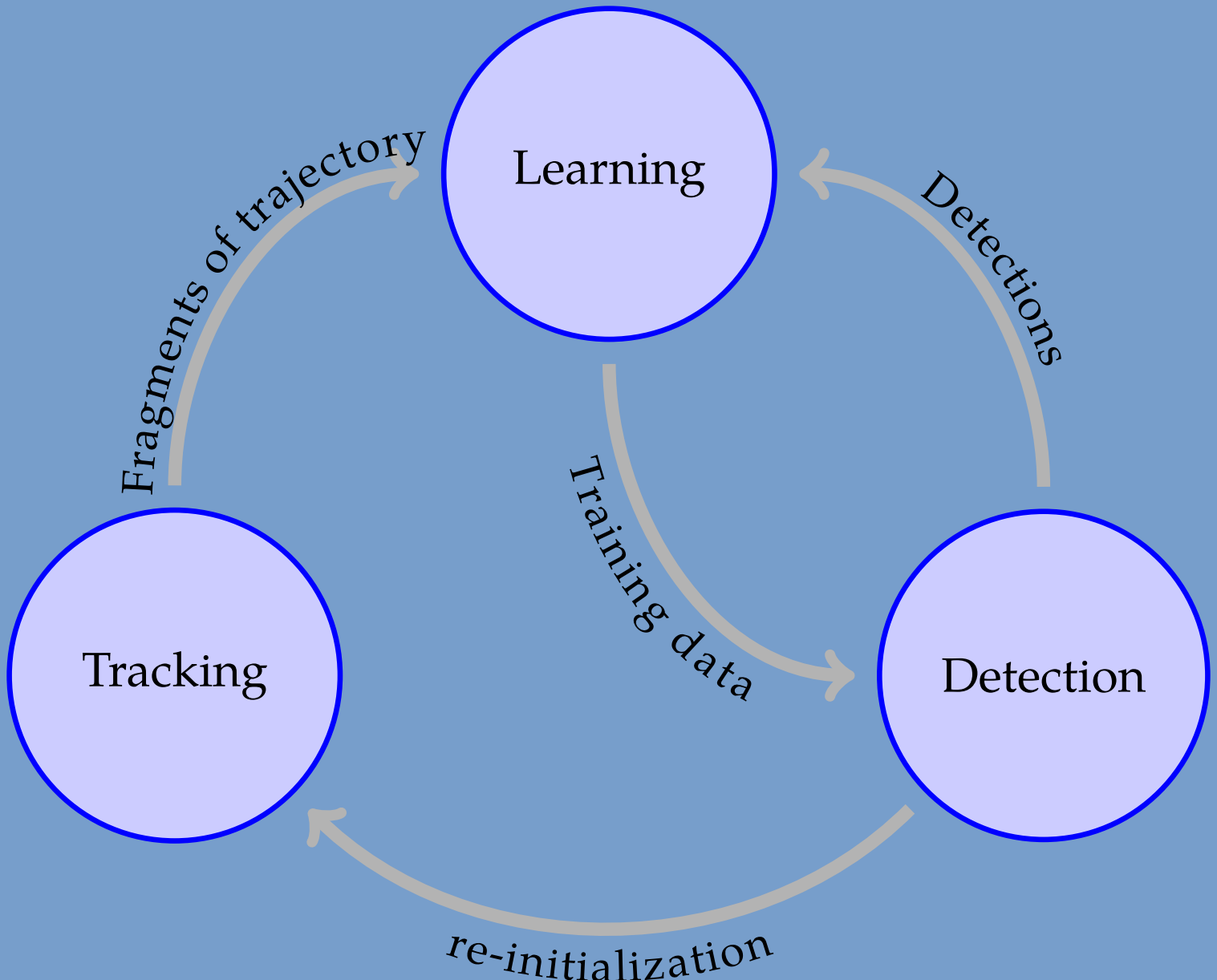
TLD Approach[2]



TLD Approach[2]



TLD Approach[2]



Improvements to Predator

Improvements to Predator

- Relatively bad tracking algorithm
-

Improvements to Predator

- Relatively bad tracking algorithm
- Kernelized Correlation Filters [3]
-

Improvements to Predator

- Relatively bad tracking algorithm
- Kernelized Correlation Filters [3]
-

Algorithm	feature	Mean precision	Mean FPS
KCF	HOG [1]	73.2%	172
KCF	Raw pixels	56.0%	154
TLD		60.8%	28

-

[1] Histogram of Oriented Gradients

Improvements to Predator

- Relatively bad tracking algorithm
- Kernelized Correlation Filters [3]
-

Algorithm	feature	Mean precision	Mean FPS
KCF	HOG [1]	73.2%	172
KCF	Raw pixels	56.0%	154
TLD		60.8%	28

- Better Learning Models

[1] Histogram of Oriented Gradients

Further applications

Further applications

Further applications

Class Register?

Assitive technologies?

Further applications

Further applications

Further applications

Deliverables

Deliverables

Deliverables

- Full Implementation of Predator in C++
  - 3rd week of 2nd term

Deliverables

- Full Implementation of Predator in C++
  - 3rd week of 2nd term
- Integrating KCF into Predator
  - End of 2nd term

Deliverables

- Full Implementation of Predator in C++
  - 3rd week of 2nd term
- Integrating KCF into Predator
  - End of 2nd term

Deliverables

- Full Implementation of Predator in C++
  - 3rd week of 2nd term
- Integrating KCF into Predator
  - End of 2nd term
- Reviewing Implementation
  - 1st week of 2nd Semester

Deliverables

- Full Implementation of Predator in C++
  - 3rd week of 2nd term
- Integrating KCF into Predator
  - End of 2nd term
- Reviewing Implementation
  - 1st week of 2nd Semester

Deliverables

- Full Implementation of Predator in C++
  - 3rd week of 2nd term
- Integrating KCF into Predator
  - End of 2nd term
- Reviewing Implementation
  - 1st week of 2nd Semester
- Improving Implementation
  - 2nd week of 2nd Semester

Deliverables

- Full Implementation of Predator in C++
  - 3rd week of 2nd term
- Integrating KCF into Predator
  - End of 2nd term
- Reviewing Implementation
  - 1st week of 2nd Semester
- Improving Implementation
  - 2nd week of 2nd Semester

Deliverables

- Full Implementation of Predator in C++
  - 3rd week of 2nd term
- Integrating KCF into Predator
  - End of 2nd term
- Reviewing Implementation
  - 1st week of 2nd Semester
- Improving Implementation
  - 2nd week of 2nd Semester
- Extension to Multiple Object Tracking
  - Beginning of 4th Term

References

[1] Z. Kalal, "Predator - tracker that learns and improves," Youtube. (2011), [Online]. Available: `https://www.youtube.com/watch?v=1GhNXHCQGsM`.

[2] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, pp. 1409–1422, 7 2011.

[3] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, pp. 583–596, 3 2014.