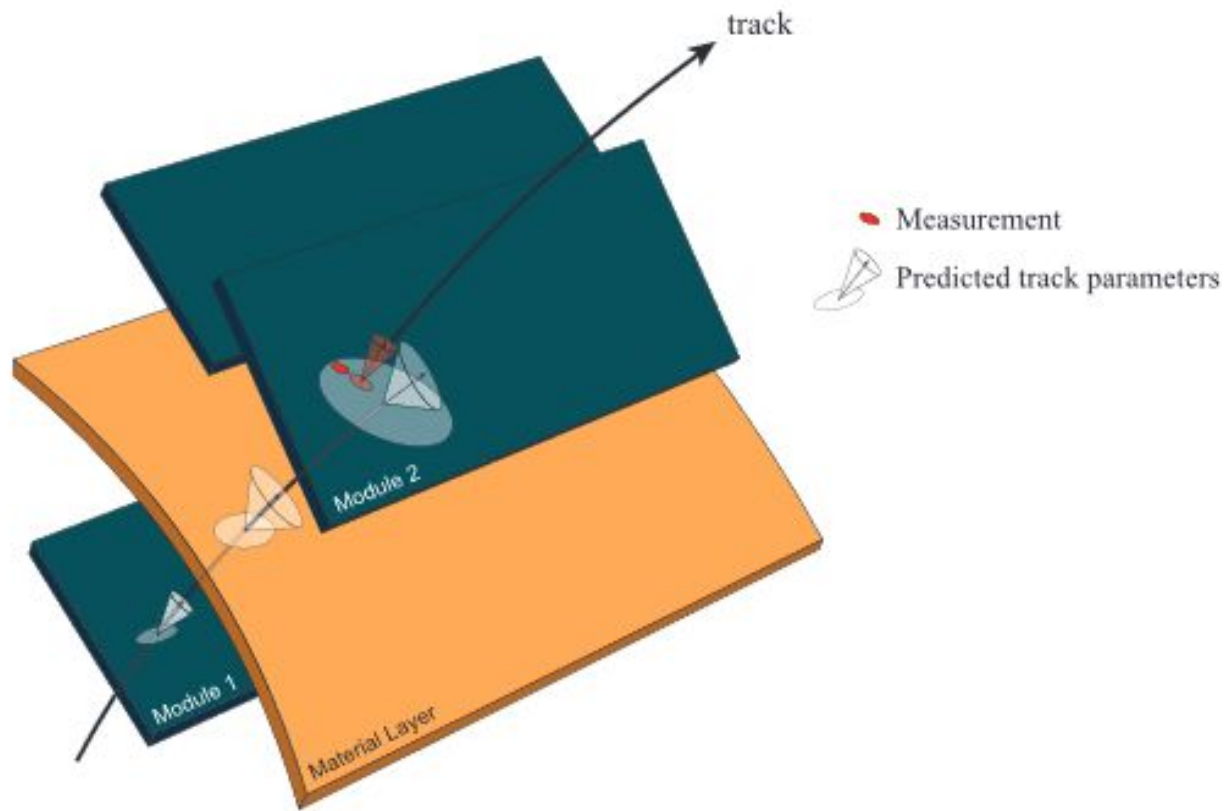
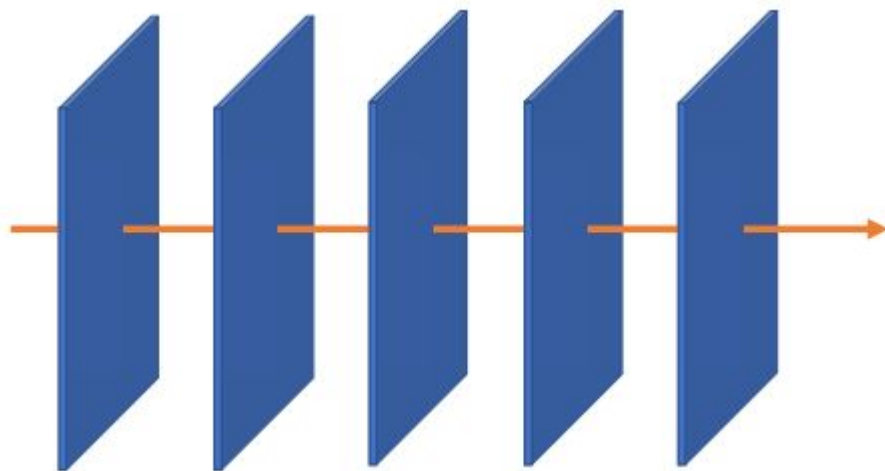
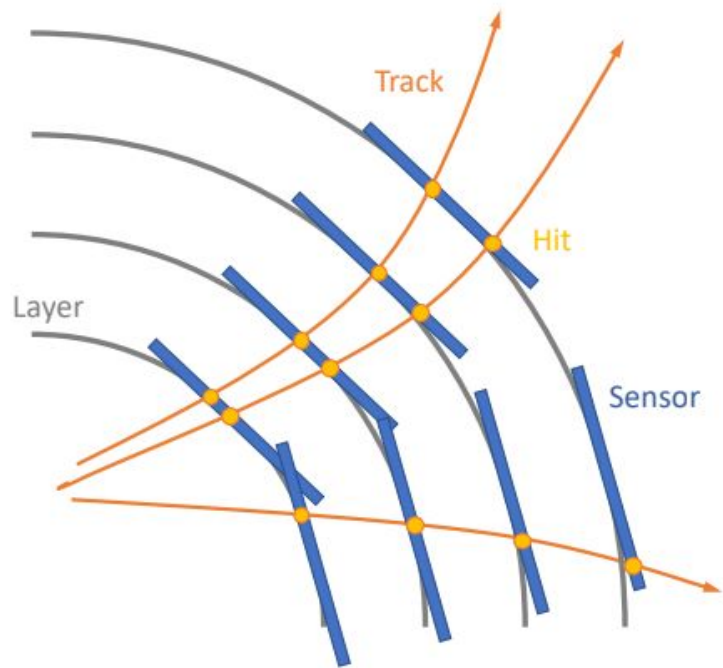


Kalman Filter in Rust

Brooks K

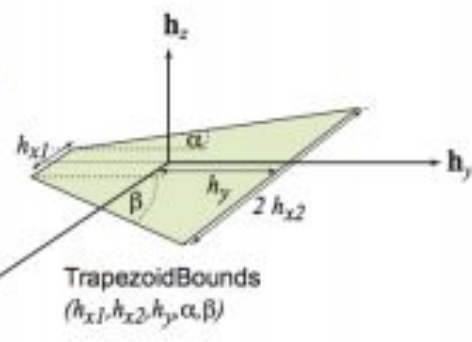
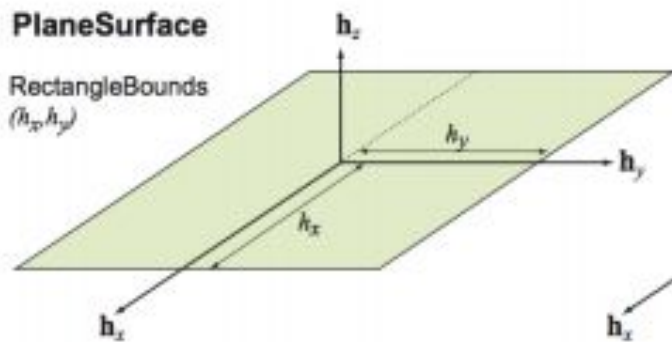
Mentors: Paul Gessinger, Hadrien Grasland, Andreas Salzburger





PlaneSurface

RectangleBounds
(h_x, h_y)



TrapezoidBounds
($h_{x1}, h_{x2}, h_y, \alpha, \beta$)

Rust

- Research language out of Mozilla
 - Originally created for Firefox
- More compile-time guarantees than C / C++
 - Very similar performance
- Easier to multithread with confidence
- Almost entirely eliminates memory management errors
 - Double free
 - Dangling pointer
 - Use after free

Kalman Filtering

- Prediction (P)
 - Based on previous data, where does the particle hit the next sensor
- Filtering (F)
 - Compare the prediction to the actual measurements
- Smoothing (S)
 - Sensors towards the end of the track will have more data to base predictions / filtering
 - Move backwards from the final sensor to the first sensor
 - Update all filtered values
 - All sensors will share the measurement data, even if they are chronologically before other sensors

P1 -> F1 -> P2 -> F2 -> P3 -> F3 -> S3 -> S2 -> S1

Key Problems

- Genericity over sensors
 - Different sensors have different ways of predicting the particle intersection
 - cylindrical / conical sensor vs planar sensor
- Reducing foreign function call (FFI) overhead from c++ to Rust
- Create rust bindings through C to a potential c++ codebase (ACTS)
 - Awful, awful, compiler errors

Further Reading

- GSoC Proposal
 - <https://summerofcode.withgoogle.com/dashboard/project/5790049365393408/details/>
- Repo
 - <https://github.com/acts-trk/rust-kf>
- Email
 - brookskarlik@gmail.com
 - paul.gessinger@cern.ch