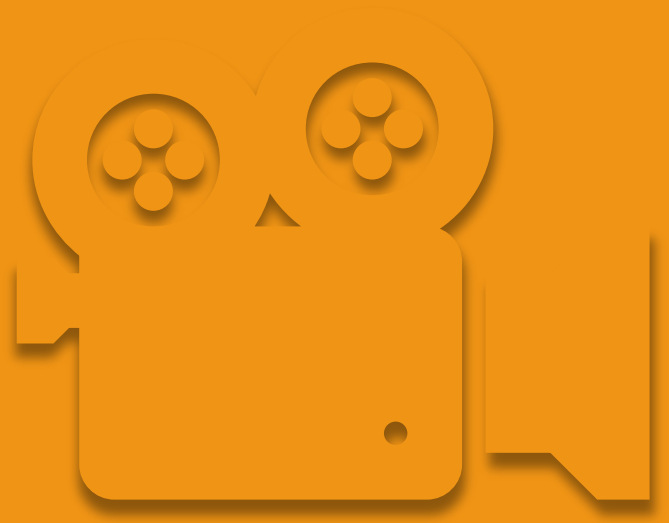
Presentation video URL:

https://youtu.be/VGenneyB-IU



Apollo Proposed Feature Presented by – 40 Below

- Team Leader - Joshua Kouri
  - Affected Modules and Directories, Lessons Learned
- Presenter - Alex Nechyporenko
  - SAAM Analysis, Presentation
- Presenter - Josh Tremblay
  - Functional Requirements, NFRs, Presentation
- Kennan Bays
  - Sequence Diagrams, Team Issues
- Mason Choi
  - Abstract, Introduction, Proposed Features, Concurrency, Conclusion

Proposed Feature - Functionality

- Feature to learn from unexpected obstacles with a new modules called “Learning”
- Submodules:
  - **Filter** - skims data outputted from perception and flags any unexpected objects. Will request information from Library as well
    - Objects wouldn’t be tracked if they aren’t recurring or if user has opted out
  - **Library** - records information from Filter and information about factors such as the time, season, and location this incident occurred at
  - **Sender** - will send all relevant information to the Scenario Handlers when needed
- New classifications of obstacles introduced:
  - **Road-Based:** occurs as a result of tire traction with the road. Ex: potholes, icy patches
  - **Obstacle-Based:** viewed as perceived obstacles. Ex: Construction work, obstructively parked cars
  - **Movement-Based:** noted by their sudden or frequently occurring changes in movement. Ex: people, poorly driven cars

Proposed Feature NFRs

Maintainability: Consistent with existing obstacles. Failures lead to safety risks for users, and downtime should be reduced

Evolvability: Evolves well as new classification and obstacles could be implemented whenever needed. Over time, they could change or additional ones would be implemented.

Performance: Same standards as existing obstacles. Quick response from all modules allow the vehicle to react to obstacles quickly. With long responses, potential collisions may occur

Security: Need methods to prevent security attacks for each obstacle, and their triggers and vehicle reactions. Also need to protect ability to enable and disable certain obstacles as it can create some risks

Testability: All obstacles should be tested to ensure correct functioning. Test that perception provides all unexpected obstacles to learning module and storytelling can create stories from each

Comparing “Learning Module” Implementation Approaches

Approach #1

- Focuses on concurrency, runs concurrently to
- Receives data from Perception at the same time as Prediction module
- Logs objects deemed “unexpected”
- Warns Storytelling module of upcoming “unexpected” objects as it returns to the area it occurred in
- Removes incidents which have not recurred
- Does not receive feedback from the Storytelling module

Approach #2

- Focuses on responsive learning
- Implemented alongside the Storytelling and Planning modules
- Records incidents but also interacts with the Scenario Handler modules to improve Learning outcomes
- Logs objects deemed “unexpected”
- Removes “unexpected” objects which have not recurred
- Does receive feedback from the Storytelling module but slower performance

Concurrency / Team Issues

Concurrency

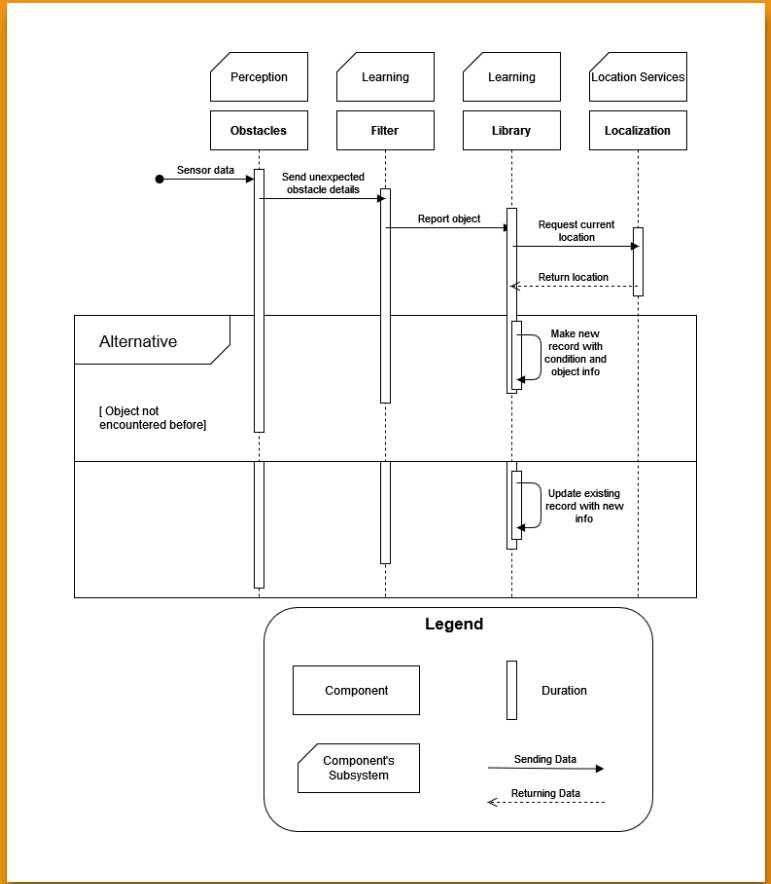
- Most of Apollo's modules run concurrently while the centralized system of Cyber RT manages them, and our proposed feature would be no different
- This is crucial for the functionality of the system as the learning module should be able to work in parallel with all the others, without the concern of a learning module failure severely impeding other modules and resulting in catastrophic failure of the entire system
- If the learning module is impeded, the Storytelling module will not be able to use its output, but it can continue to do the rest of its tasks

Team Issues

- Team with history of AI experience or hire a consultant with domain experience that is relevant to the Learning module for AI
- It would also be important to work with or take on developers from the Storytelling module, due to the Learning module's integral interactivity with it

Sequence Diagram - Unexpected Obstacle Encountered

- Perception module sends unexpected obstacle details to the Filter submodule
- Filter submodule reports obstacle data to Library submodule
- Library submodule queries Localization for the current location, then checks if this obstacle has been encountered before:
  - If this obstacle is new, Library creates a record with the conditions and object information
  - If this object has been encountered before, Library appends the new information to the existing entry



SAAM Analysis - Part 1

- Scenario:
  - Create a component that allows for dynamic learning based on unexpected encounters on the road, be it weather based, environmental, or traffic conditions
  - Can enable a safer driving experience and robustness via prediction handling
- Stakeholders:
  - Apollo Driver
  - Development Team
  - Baidu (Parent Company)
  - Baidu Shareholder

SAAM Analysis - Part 2

- From a maintainability standpoint, approach 1 is stronger due to it being more stable and consistent in its communications with other components
- In contrast, approach 2 is more dependent on other subsystems, therefore, will struggle more with maintenance and evolvability
    - The greater range of interactivity and dependence will affect more within the software system, therefore making it difficult to expand on the current feature or to test/maintain the existing software.
- From a performance perspective, while approach 2 will be slower the difference is largely negligible
    - Approach 1 will be more efficient due to concurrency being a key component of it, where it acts as an intermediary of Perception and Prediction modules

SAAM Analysis - Part 3

- Due to the increased interactivity of approach 2, it will introduce certain vulnerabilities to the code, however, from a perspective of the safety of the driver it will actually increase
    - The increased interactivity between modules will enable the learning module to better adapt and generate different responses than approach 1
    - This however, will require more effort from an implementation standpoint to build effectively
- Approach 1, on the other hand, has a well defined edge over approach 2 in evolvability due to its isolated nature
    - Due to its reduced impact on surrounding modules, more changes can be made to approach 1 of the module with added new extensions to the feature
    - In contrast, approach 2's interactivity will make changes more difficult due to its potential implications on other subsystems as a consequence
- Lastly, from a testability perspective, due to approach 1 acting more as an intermediary between Perception and Prediction modules, it is more localized with fewer interaction. This Makes it much easier to test in contrast to approach 2

SAAM Analysis - Conclusion

- Approach 1 is easier to implement, is more cost effective and gets the desired results
  - Due to the localization nature of approach 1, the module is easy to streamline and to maintain standardized measure for testing and evolving the code base for the feature
- That being said, approach 2 is very secure and can handle more responses to different encounters on the road
  - From an NFR and stakeholder viewpoint, security from a code and road safety is extremely important. Safety impacts the branding of the parent company Baidu, as well as the integrity of the data of the software
- The question ultimately comes down to if security of this module is more important than the other benefits of approach 1
- Ultimately, approach 1 is still the desired option, due to its cost effectiveness and easy of implementation, along with it greater ability to evolve.
  - Evolvability enables Apollo to develop new additions to this feature, which can eventually make approach 1 a more safe option from a driving perspective than approach 2

Affect Directories

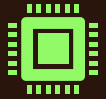
Perception, Prediction, and Scenario Handlers

Dependencies between Learning module must be introduced

Support for “Expected” obstacles must be introduced

Storytelling

Storytelling must be reworked to factor the output of the Learning module into its decisions



Monitor

An interface to enable/disable learning relating to obstacles, such as ice patches, is introduced

Location Services

Dependency is introduced to from Learning module to Location Services to get location information relevant to recording

Lessons Learned and Limitations

- Were not able to find any public reference architectures which would detail how our proposed architecture would work when coinciding with our feature
- Many potential options for implementing this feature via placing the module within different sections of the architecture, along with potential interactivity it can have with other subsystems
- Our team learned how to evaluate the effectiveness of different approaches long-term. By prioritizing modularity and independence, despite lacking experience in this domain managed to create flexible time analysis and development
- From a SAAM Analysis perspective, it is difficult to make concrete decisions based on analysis of which NFRs would better serve stakeholders when contrasting the two approaches. We learned how to effectively compare by weighing non-comparable attributes

Conclusion

- To summarize, our proposed feature would deal with the issue of encountering unexpected obstacles by introducing a learning module to deal with them
- Two possible approaches were generated for implementing this feature (locating it between the perception and prediction modules, or locating it with the scenario handler modules)
- after performing a thorough SAAM analysis, we have concluded that the first approach would be the best way for this feature to be implemented due to easy of implementation and cost effectiveness, along with it maintaining NFRs that align with stakeholder needs
- Overall, we think that this would be a beneficial feature for Apollo to have