Evaluation of ALC Toolchain on Neural Network models and Architectures

Hunter Wang

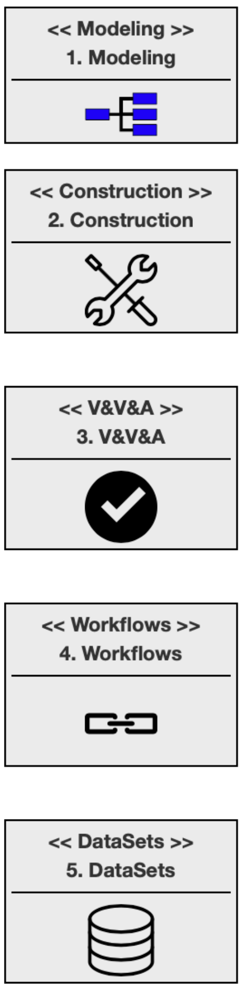
Mentor: Shreyas Ramakrishna
P.I. Dr. Gabor Karsai

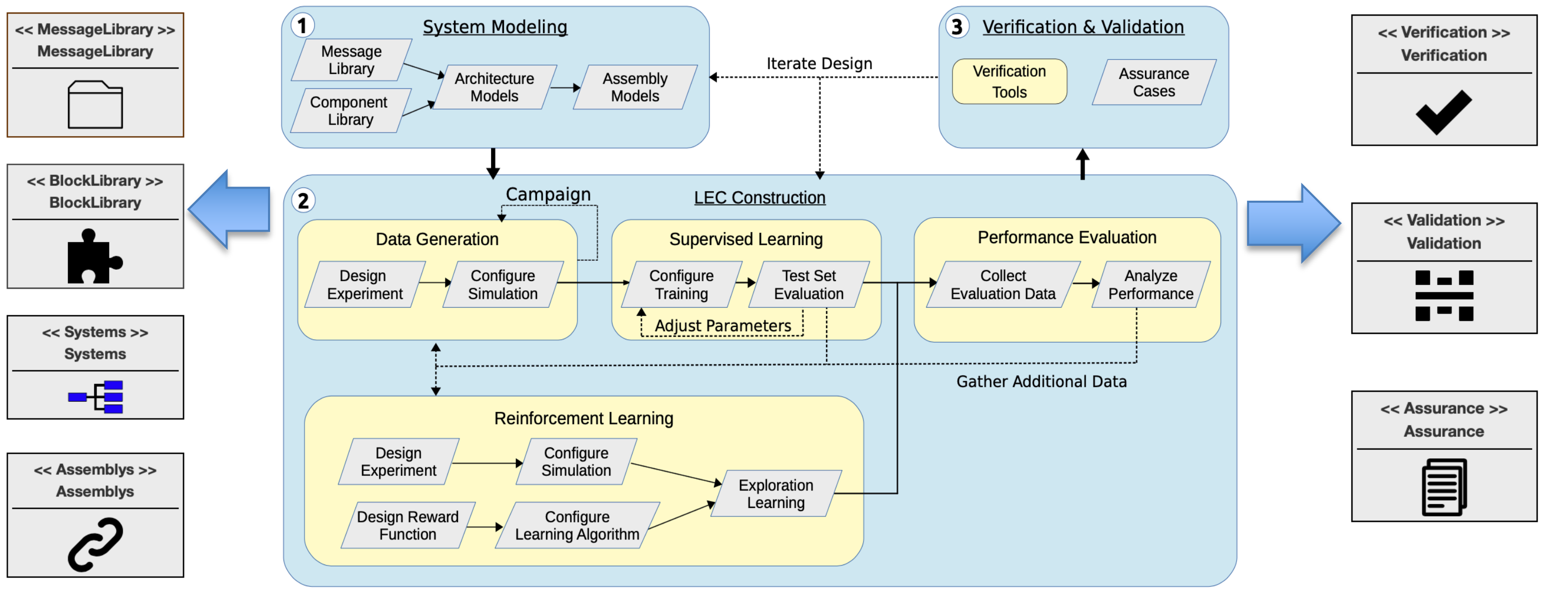
ISIS

Tel (615) 343-7472 Fax (615) 343-7440
1025 16th Avenue South|Nashville, TN 37212
www.isis.vanderbilt.edu

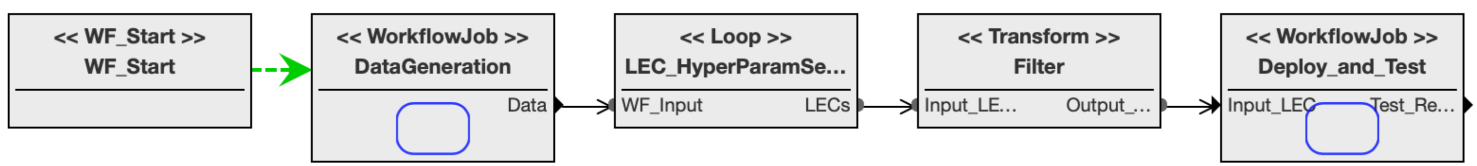
VANDERBILT UNIVERSITY

Background – ALC Toolchain

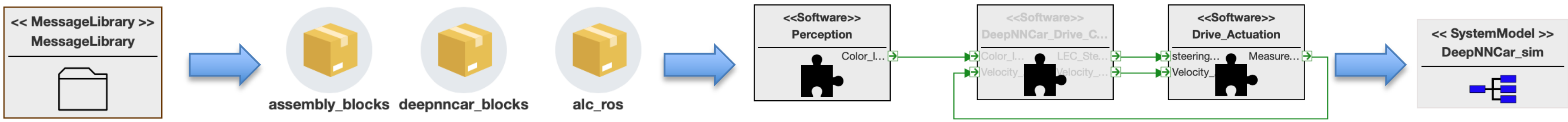




Source: Charles Hartsell, Nagabhushan Mahadevan, Shreyas Ramakrishna, Abhishek Dubey, Theodore Bapty, Taylor Johnson, Xenofon Koutsoukos, Janos Sztipanovits, and Gabor Karsai. 2019. CPS Design with Learning-Enabled Components: A Case Study. In *Proceedings of the 30th International Workshop on Rapid System Prototyping (RSP'19)* (RSP '19). ACM, New York, NY, USA, 57-63.



ALC Toolchain for Autonomous Driving



Message Library

Block Library

System Model

Deploys ROS packages with ROS nodes onto the server running the ALC toolchain

Assembly Model

Dataset Collection

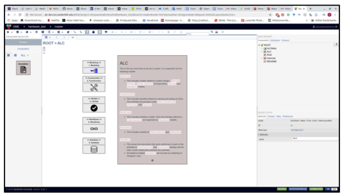
LEC Training

LEC Testing

LEC construction elements

System modeling → ROS package generation → LEC construction & training → Deploying toolchain generated files → Continual learning

ALC Toolchain



Output

1. Top level ROS launch file.
2. Component ROS package with ROS nodes and launch files
3. Trained LEC weights.

Deploy

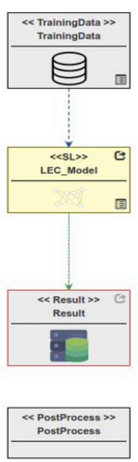
1. System modeling, ROS node generation, LEC construction and training for DeepNNCar

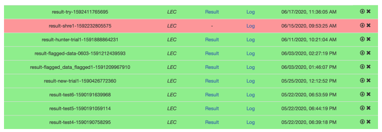
Training Data Selection

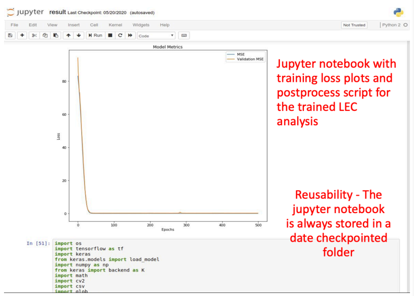
LEC model design

Analysis of trained model

Model evaluation and loss plotting

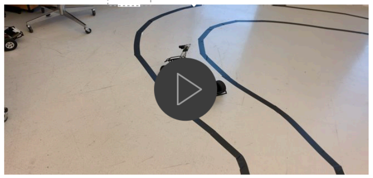






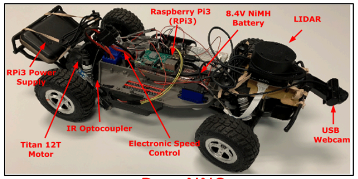
Jupyter notebook with training loss plots and postprocess script for the trained LEC analysis

Reusability - The jupyter notebook is always stored in a date checkpointed folder



end-to-end image to steering control

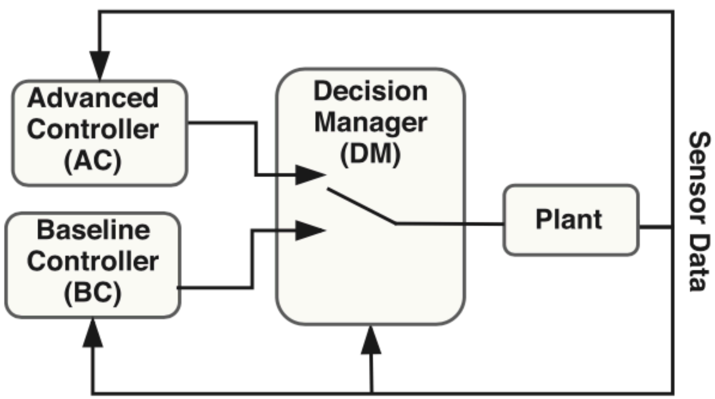
Run

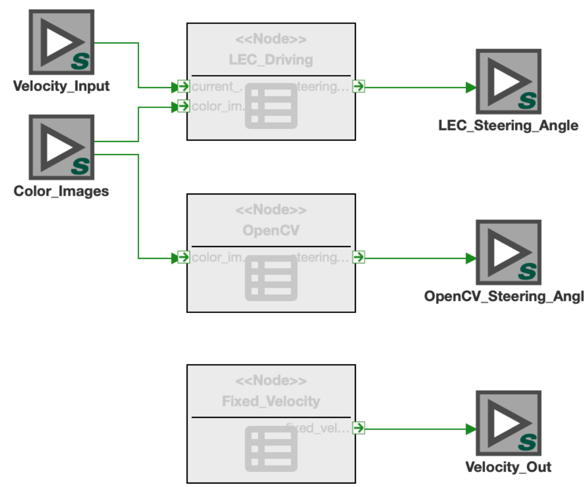


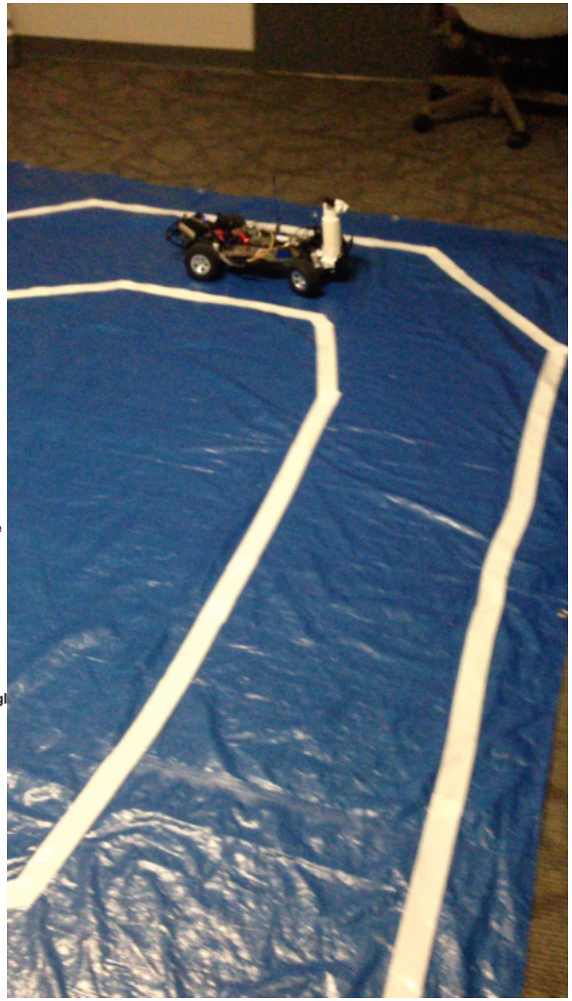
DeepNNCar

Simplex Strategy Implementation on ALC Toolchain

- Goal: Evaluate the the strength of the ALC Toolchain when applied for modeling an autonomous car setting
- Fixed-weight simplex strategy to compute the steering angle
- Two controllers: LEC_Driving and OpenCV



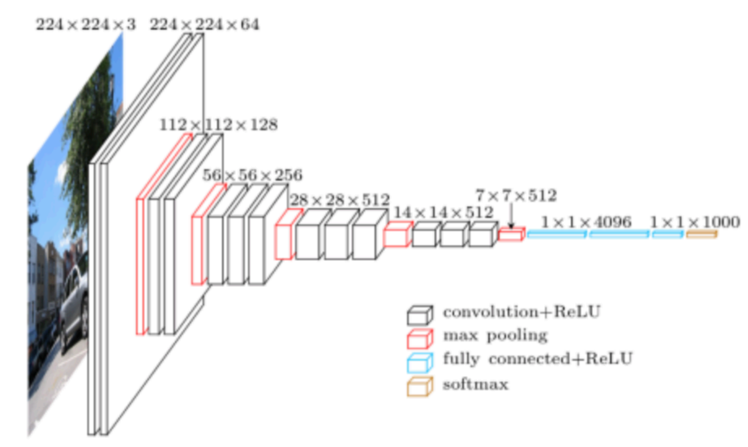




Source: Shreyas Ramakrishna, Charles Harstell, Matthew P. Burruss, Gabor Karsai, Abhishek Dubey. Dynamic-weighted simplex strategy for learning enabled cyber physical systems. Journal of Systems Architecture, Volume 111, 2020, 101760, ISSN 1383-7621,

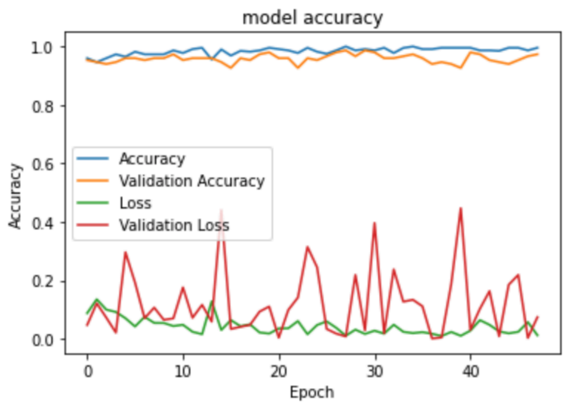
VGG16 Model Implementation on Google Colab

- VGG16 Model
- 2 classes: vehicles and non-vehicles.
- The model can be used to decide when to stop
- Implemented with Keras.
- Dataset: 3425 images of vehicles and 3900 images of non-vehicles.
- Trained and tested on Google Colab.



Architecture of VGG16

Source: https://arxiv.org/abs/1505.06798



- Validation accuracy: 98%
- Testing accuracy: 96.67%
- Lack of reproducibility
- Lack of traceability

Summary

- Future work:
  - Integrate the Dynamic-weighted Simplex Strategy into the Toolchain
  - Improve the training and testing experience within the Toolchain
- What I learned:
  - ALC Toolchain allows for collaboration between multiple users while maintaining reproducibility and traceability during all stages of the development cycle of a CPS system
  - An automated modeling environment can save a lot of time
- Challenges:
  - Work and communicate remotely