Vision Dynamics for controlling autonomous vehicles

Prof. Sorin Grigorescu

Elektrobit Automotive Romania
Transilvania University of Brasov, Romania

World Usability Day
November 12, 2020
About Elektrobit (EB)

Technical competencies
EB’s technical core competencies are development of automotive-grade (software) products and engineering services.

Employees
More than 2600 employees* worldwide. Spans three continents and eleven countries.

Consistent growth
In 2018: +35 %

Global presence
Development and business offices in Austria, China, Finland, France, Germany, India, Israel, Japan, Romania, South Korea, and USA.

Continental AG
Wholly owned, independent subsidiary of Continental AG.

100+ million
Over 100 million vehicles on the road and 1 billion embedded devices.

*December 2018, incl. Argus, excl. e.solutions.
Scientific Achievements: Visual Control of Robots

- **From visual perception to motion planning and control**
- **Applications**: Self-driving cars (Elektrobit Automotive), FRIEND rehabilitation robot (University Bremen), PR2 service robot (Willow Garage, Google)

Visual Data Acquisition
- Stereo camera
- Time-of-Flight
- Structured light (MS Kinect)
- LiDAR

Pose Estimation → 3D Reconstruction and Object Tracking → Scene Understanding → Motion Planning and Control
Computer Vision vs. Visual Robotic Control

Computer Vision Task

Visual Robotic Control Task
Robotic Systems as Intelligent Agents

Driving Environment
- Highway
- Inner-city
- Parking place

Video Cameras
- GPS
- LiDAR

Control actions
- Observation
- Acceleration
- Steering control

Radars
- IMU

Autonomous Vehicle Hardware Setup
- Quanergy M8 LiDARs
- Continental ARS430 Radars
- Continental MFC430 Front Camera
- Differential GPS

Autonomous Vehicle Control Pipeline

Perception-Plan-Action processing pipeline

Deep Learning (or Classical) Perception and Localization → Deep Learning (or Classical) High-Level Path Planning → AI-based (or Classical) Behavior Arbitration (low-level path planning) → Learning-based (or Classical) Motion Controllers → Autonomous Vehicle

Safety Monitor

End2End learning for direct sensors-to-actuation mapping

End2End Learning → Autonomous Vehicle

Safety Monitor

Better Learning-based Modules through AI

Artificial intelligence (AI) is intelligence exhibited by *machines*.

- “Artificial Intelligence” describes a machine that *mimics human cognitive functions*, such as learning and problem solving
- Currently dominated by *Machine Learning* and *Deep Learning* (large scale statistical learning systems)

- General programming language
- Neurons arrangement for forming concepts
- Higher human brain functions simulation
- Abstraction: deal with ideas
- Creativity and randomness
- Self-improvement
- Measure problem complexity
- Learning-based modules
- Non-learning modules
Traditional *Perception-Plan-Act* Pipeline

Perception and Control components are treated independently of each other

- Perception and path planning are decoupled from the motion controller
- **Advantages:** reduced design complexity due to modularization
- **Disadvantages:**
  - Disturbances and intrinsic components dependencies are not taken into account
  - If one component fails (e.g. path planning), the entire control system will fail

Environment Perception (world modelling) → Behavioral Planning → Path Planning (e.g. A* or DWA) → Motion Controller → Autonomous Vehicle

\[
\begin{align*}
\text{Environment Perception (world modelling)} & \rightarrow \text{Behavioral Planning} \rightarrow \text{Path Planning (e.g. A* or DWA)} \rightarrow \text{Motion Controller} \rightarrow \text{Autonomous Vehicle} \\
& \rightarrow \text{Vehicle state}
\end{align*}
\]
Traditional *Perception-Plan-Act* Pipeline

Perception and Control components are treated independently of each other.

**Visual Perception Design**
- **Computation Intelligence** community
- **Goal**: transform the environment in a machine understandable form

**Control System Design**
- **Automatic Control** community
- **Goal**: compute optimal control actions, based on a virtual representation of the environment

Diagram:
- Environment Perception (world modelling)
- Behavioral Planning
- Path Planning (e.g. A* or DWA)
- Motion Controller
- Autonomous Vehicle
- Vehicle state
- Localization

Math:
- $z^d_{<t+1, t+\tau_o>}$
- $u_{<t+1, t+\tau_o>}$
- $z_{<t>}$
- $z_{<t-\tau_i, t>}$
Vision Dynamics Framework

Environment perception and control based on analytical and statistical models

- Predict future observations $\hat{x}_{<t+1,t+\tau_o>}$, state trajectories $z_{<t+1,t+\tau_o>}$ and control inputs $\hat{u}_{<t+1,t+\tau_o>}$
- Optimize control inputs over prediction horizon $[t + 1, t + \tau_o]$

• Assumptions:
  - The observations, states and actions are continuous and sampled at discrete time
  - The dynamics are governed by the physical laws of classical mechanics

Vision Dynamics Approach

Environment perception and control based on analytical and statistical models

• Predict future observations $\mathbf{x}_{<t+1,t+\tau_o>}$, state trajectories $\mathbf{z}_{<t+1,t+\tau_o>}$ and control inputs $\mathbf{u}_{<t+1,t+\tau_o>}$

• Optimize control trajectories over prediction horizon $[t+1, t+\tau_o]$
Vision Dynamics Approach

State transition modelling

given (a set of observations $x$, states $s$ and actions $a$)

$$x^{<t-\tau_i,t>}, s^{<t-\tau_i,t>}, a^{<t+1,t+\tau_o>}$$

find (a mapping)

$$h: X \times S \rightarrow A$$

such that

$$z^{<t+1>} = f(z^{<t>}, u^{<t>}) + h(s^{<t-\tau_i,t>})$$

$	au_i$ – past sampling time horizon (input)

$	au_o$ – prediction horizon (output)

encodes the scene’s temporal dynamics

$a$–priori model

learned statistical model
Autonomous Driving Problem

A vision dynamics control perspective

• **Given**
  – a sequence of past occupancy grid observations
    \[ X_{<t-\tau_i,t>} = [x_{<t-\tau_i>}, ..., x_{<t-1>}, x_{<t>}] \]
  – the position of the ego-vehicle \( p_{ego_{<t>}} \in \mathbb{R}^2 \) in occupancy grid space \( x_{<t>} \)
  – and a destination position \( p_{dest_{<t+\tau_o>}} \)

• **the task is to**
  – estimate a local state trajectory \( Y_{<t+1,t+\tau_o>} = [y_{<t+1>}, ..., y_{<t+\tau_o>}] \) (yellow line), encoding position and velocity
  – to destination point \( p_{dest_{<t+\tau_o>}} \)
  – along a prediction horizon \( \tau_o \)

• **Observations:**
  – sequences of past occupancy grids \( X_{<t-\tau_i,t>} \), marking free-space with green and obstacles with red
Sequences of Occupancy Grids as Observations

- Occupancy Grids are fused video camera, LiDAR and radar data observations (green: free space, red: obstacles, black: unknown)

Simulated Occupancy Grids

Real-world Occupancy Grids
Representation Learning in Robotic Perception

- Laser
- GPS
- IMU
- Camera

Grid Fusion Space → Representation Learning → Deep Neural Networks

- AI Inference Engine A
- AI Inference Engine B
- AI Inference Engine C
- AI Inference Engine n

- Boundaries and lanes detector
- Driving environment recognition
- Drivable area segmentation
- Motion planning and control
Deep Grid Net (DGN)

Deep Grid Net (DGN): A Neural Network for Driving Context Understanding

• **Input:** Grid data acquired from different driving scenarios
• **Output:** Driving context information provided as a three classes probabilistic output (inner city driving, motorway driving and parking)

Deep Grid Net (DGN)
**NeuroTrajectory**: Perception-Planning Deep Network

- Local state prediction
- Encode environment perception, path planning and behavioral planning within a single statistical model (DNN)

**Vision Dynamics Model**
(Environment Perception; Path Planning; Behavioral Planning)

\[ Y_{<t+1,t+\tau_0>} \]

**Constrained Nonlinear MPC**
\[ u_{<t+1,t+\tau_0>} \]

**Localization**
\[ \hat{z}_{<t-\tau_i,t>} \]

**Autonomous Vehicle**
\[ z_{<t>} \]

**Vehicle state**

NeuroTrajectory: Vision Dynamics Model

- CNN<\(t-\tau_i, t\)> sequence of spatial features is further fed to a stack of LSTM branches
- \(\mathbf{y}_{<t+1, t+\tau_0>} = [\mathbf{y}_{<t+1>}, \ldots, \mathbf{y}_{<t+\tau_0>}\) - estimated local states along prediction interval \([t + 1, t + \tau_0]\)
- \(\mathbf{y}_{<t+i>}\) - local state element predicted by LSTM branch \(i\)
**NeuroTrajectory: Evolutionary DNN Training**

- Multi-objective training on *Objective Space L*: evolving a population of deep neural networks $\varphi_K$
- Multi-objective loss: the ego-vehicle’s i) traveled path $l_1$, ii) lateral velocity $l_2$ and iii) longitudinal velocity $l_3$

*M = (S, A, T, L)*

- $S$ – states trajectories $s_{<t-\tau_o>}$
- $A$ – trajectory sequences; $Y_{<t+1,t+\tau_o>}$
- $T$: $S \times A \times S \rightarrow [0,1]$ – transition function describing the probability of arriving in state $s_{<t+\tau_o>}$, after optimizing over trajectory $a_{<t>}$
- $L$: $S \times A \times S \rightarrow \mathbb{R}^n$ – multi-objective cost function quantifying the quality of trajectory $Y_{<t>}$

\[
\begin{align*}
    l_1^{<t+\tau_o>} &= \sum_{i=1}^{\tau_o} \|p_{ego_{<t+i>}} - p_{dest_{<t+i>}}\|_2^2 \\
    l_2^{<t+\tau_o>} &= \sum_{i=1}^{\tau_o} v_\delta^{<t+i>} \\
    l_3^{<t+\tau_o>} &= \sum_{i=1}^{\tau_o} v_f^{<t+i>} \in [v_{min}, v_{max}] 
\end{align*}
\]
NeuroTrajectory: Evolutionary DNN Training

- Multi-objective training on Objective Space \( L \): evolving a population of deep neural networks \( \varphi_K \)
- \( \varphi_i(\Theta_i) \) – deep network individual with weights \( \Theta_i \)
- \( \Phi^* = [\varphi_1^* (\Theta_1^*), ..., \varphi_K^* (\Theta_K^*)] \) - Pareto front of optimal deep neural networks
Car pos x: 89.11
Car pos y: 432.41
rel x: 89.11
rel y: 432.41
velocity: 0.0 km/h

Predicted delta values:
dx1: 0.0, dy1: 1.7
dx2: 0.0, dy2: 2.5
dx3: -0.1, dy3: 3.4
dx4: -0.1, dy4: 4.3
dx5: -0.2, dy5: 5.1
dx6: -0.2, dy6: 6.0
dx7: -0.3, dy7: 6.9
dx8: -0.4, dy8: 7.8
dx9: -0.4, dy9: 8.7
dx10: -0.5, dy10: 9.5
Highway Driving

velocity: 86.25 km/h
altitude: 381.8 +/- 1.61 m
latitude: 48.81 +/- 0.01 deg
longitude: 11.47 +/- 0.01 deg
pos X in WCS: 1897.12 m
pos Y in WCS: 7555.76 m

Reference trajectory
NeuroTrajectory

North
← West
South
→ East
Neural net activations: steering = 0.02, throttle = 1.0
Thank you!

World Usability Day
November 12, 2020

Sorin.Griorescu@elektrobit.com