Learning from Observation

Avelino Gonzalez
Hans Fernlund
University of Central Florida

Learning from Observation

- Young children learn much by observing adults do things.
- Some animals also learn to imitate human actions - "*monkey see monkey do*".
- Some knowledge can be difficult to articulate. Often referred to as
  - *tacit knowledge*
  - *implicit knowledge*
- It is often better to show how something is done rather than to tell about it

Application Domain

- We want to build agents that can act in a tactically correct manner
  - In a simulation
    - For training
    - For analysis
    - For fun
  - In the real world
    - To perform real tasks
- Must build a “model” of human performance to control the agent

Example Application Domains

- Conflict-based
  - Sports games (football, baseball, basketball)
  - Video games (Quake, Doom, etc.)
  - Military operations
- Non-conflict-based
  - Driving automobiles (airplanes, ships, buses, trains, etc.)

Benefits of Learning from Observation

- Makes the model building process more manageable
  - Quicker
  - Cheaper
  - Fewer errors
- Can learn models from observation of people who are unable or unwilling to assist in the process
  - Opponents in games
  - Enemy in war

Some Prior Work (1)

- Schaal, 1999 proposes work on lfo to deal with very large search spaces
- Sidani, 1994 used neural networks and rules to learn to drive a car
- Henninger, 2001 used neural nets to learn how to drive battle tank
- Moukas and Hayes, 1996 used lfo to model social behavior of honeybees

Some Prior Work (2)

- Sammut et al, 1992 uses lfo to learn to fly, but it purely imitates
- Pomerlau, 1996 uses neural networks to drive a robotic car
- Bentivegna and Atkeson, 2001, explored lfo by using primitives
- Stensrud, 2004 used FuzzyARTMap neural nets to learn context transitions in Poker

Problem Statement

To more easily develop human-like tactical agents with individual behavior pattern by implementing learning from observation

Note

- As in all learning tasks, care must be taken to avoid “imitation”
  - That is, exact replication of what is seen
- In real world (other than maybe in manufacturing), the same conditions not seen repeatedly
  - Learning must be able to generalize
- This was the challenge

Our Approach

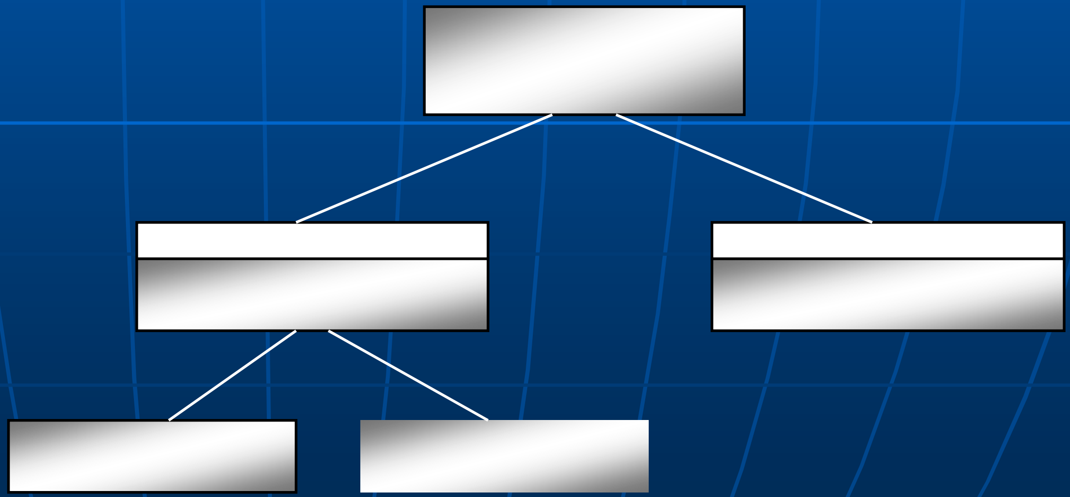
- Build models of human performance semi-automatically, with no a priori knowledge about the world *
- Developed by Dr. Hans Fernlund as his doctoral dissertation
  - PhD in Computer Engineering, University of Central Florida, May 2004
- We now discuss the model building process

Learning Models of Human Performance

- Uses Context-based Reasoning (CxBR) as the modeling infrastructure for human behavior
- Uses Genetic Programming (GP) as the learning strategy.
- Observes a human actor in simulator executing the desired mission.
- Called *Genetic Context Learning* (GenCL)
- Features Rigorous evaluation of work.

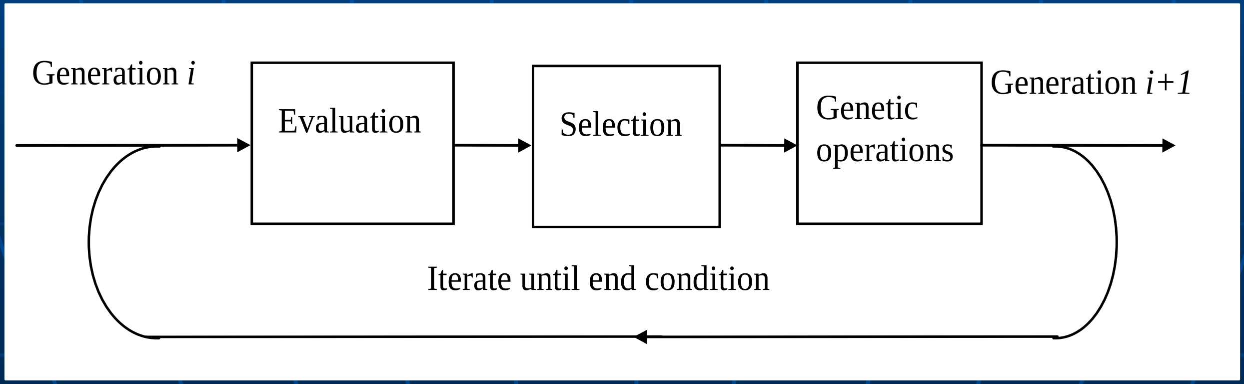
GenCL

- Context-Based Reasoning (CxBR)
  - Situational Awareness
  - Hierarchical structure
  - Limits Search Space
  - Intuitive



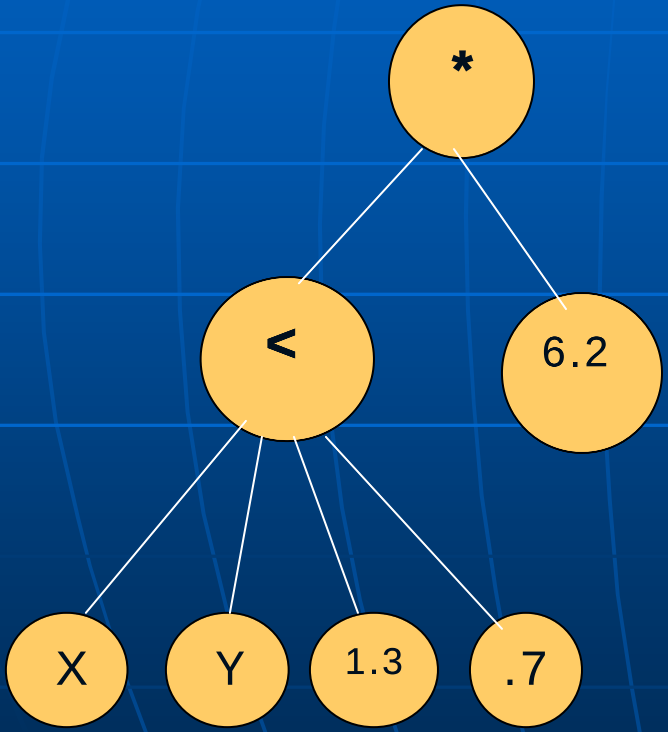
Genetic Programming

- Evolutionary Algorithm
- Generates source code from function tree
- Applicable to many problem domains
- Non-transforming



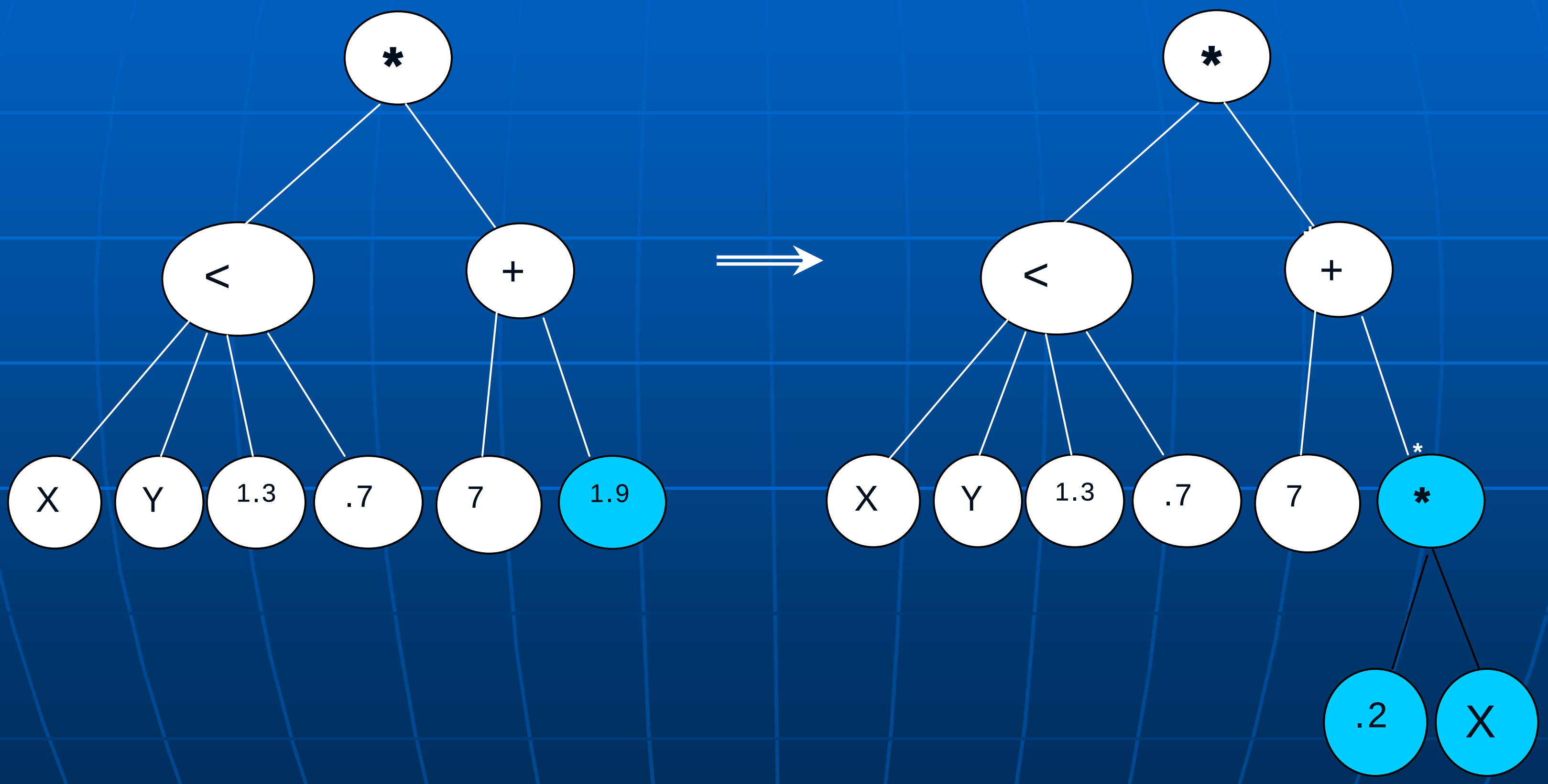
Instruction Trees

- Individuals are represented as instruction trees:

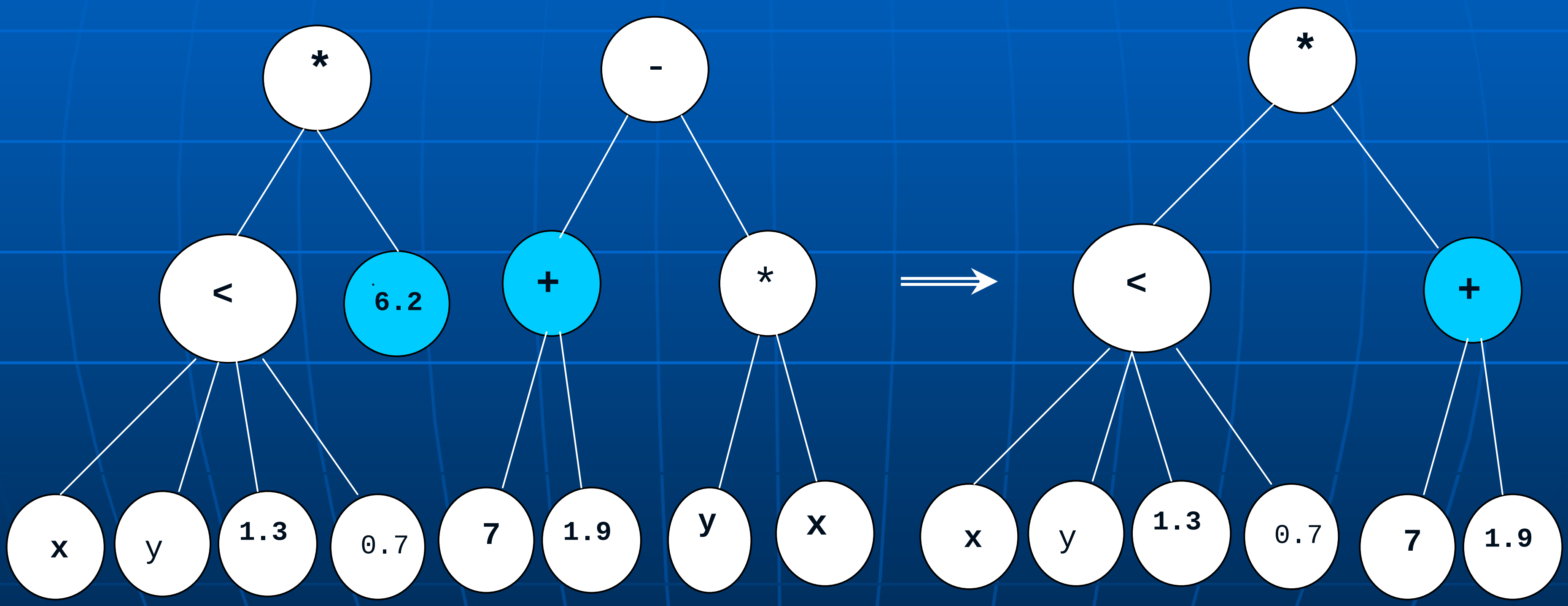



```
float foo(){
    if(x<y)
        return 1.3*6.2;
    else
        return 0.7*6.2;
}
```

Mutation



Crossover



Fitness Function

- The fitness function is a key aspect of evolutionary algorithms
- In our approach, the fitness function is the record of performance by the expert actor
- It is obtained by observing his/her performance on a simulator
  - Over the repetition of the same actions during a run
  - Over a few runs

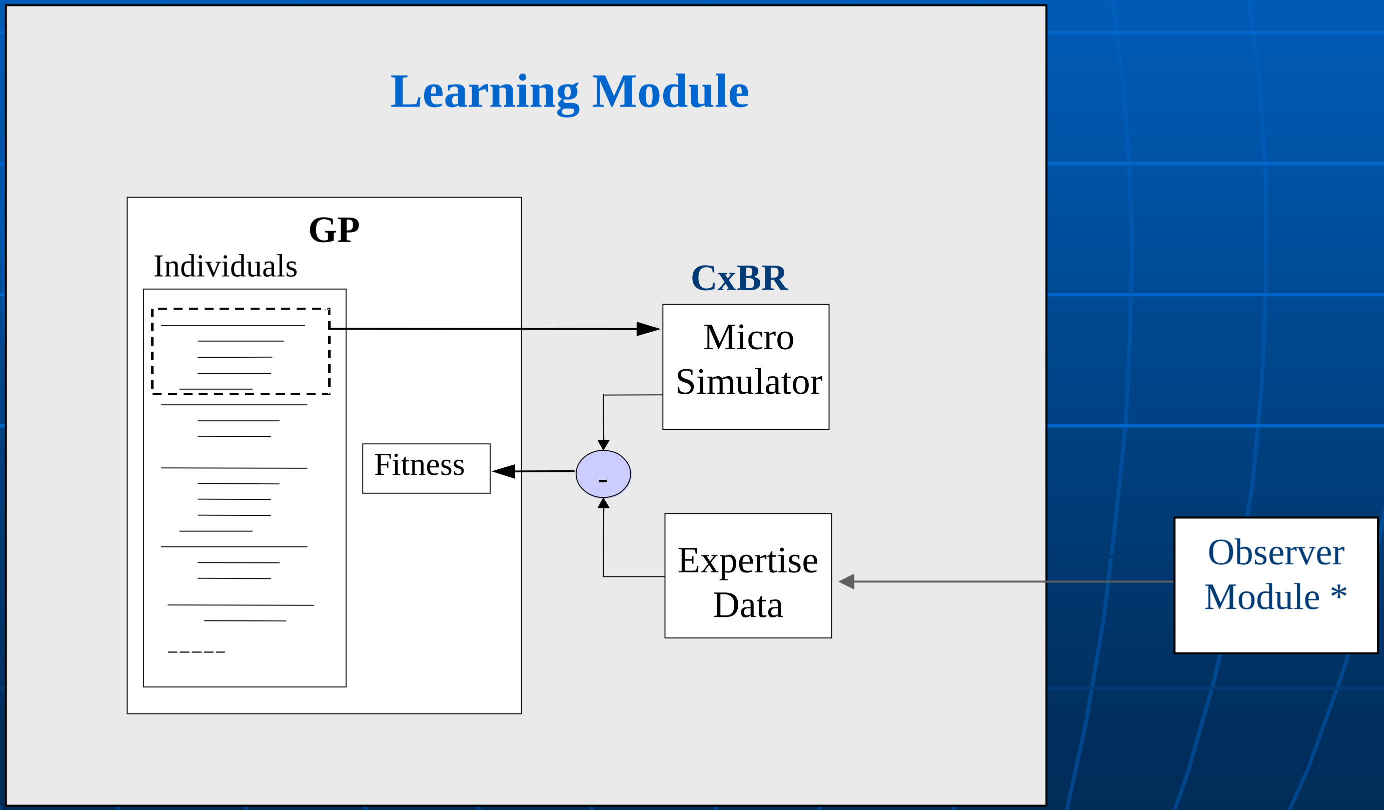
Layered Learning GP

- Bottom-up approach to leaning in a hierarchical structure
- Developed by Hsu and Gustafson in 2001
- Learn the lower level contexts first
- Places them in the function tree for higher level contexts to use
- Fits very well with the naturally hierarchical structure of context-based reasoning

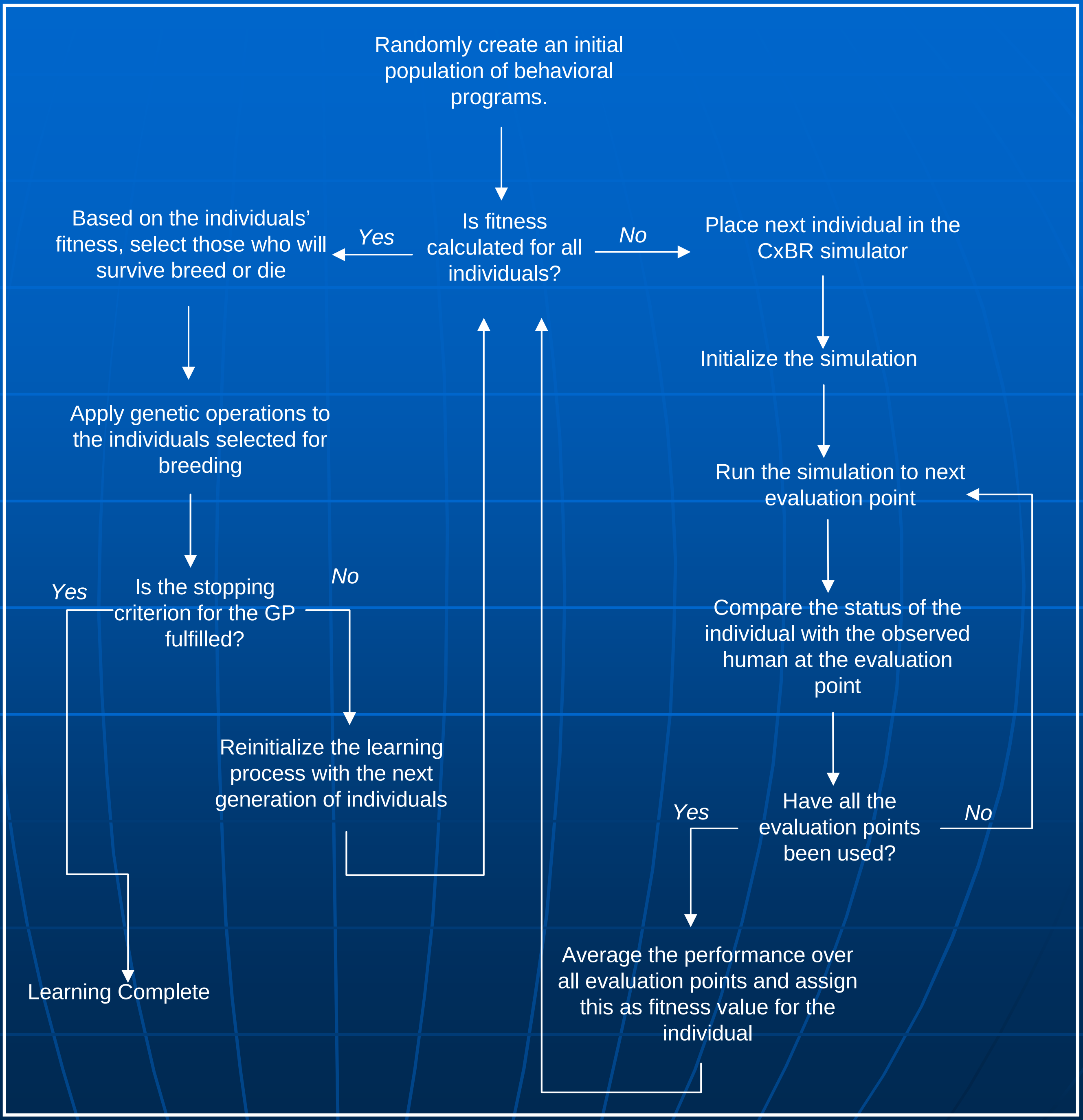
Cooperative Co-Evolution

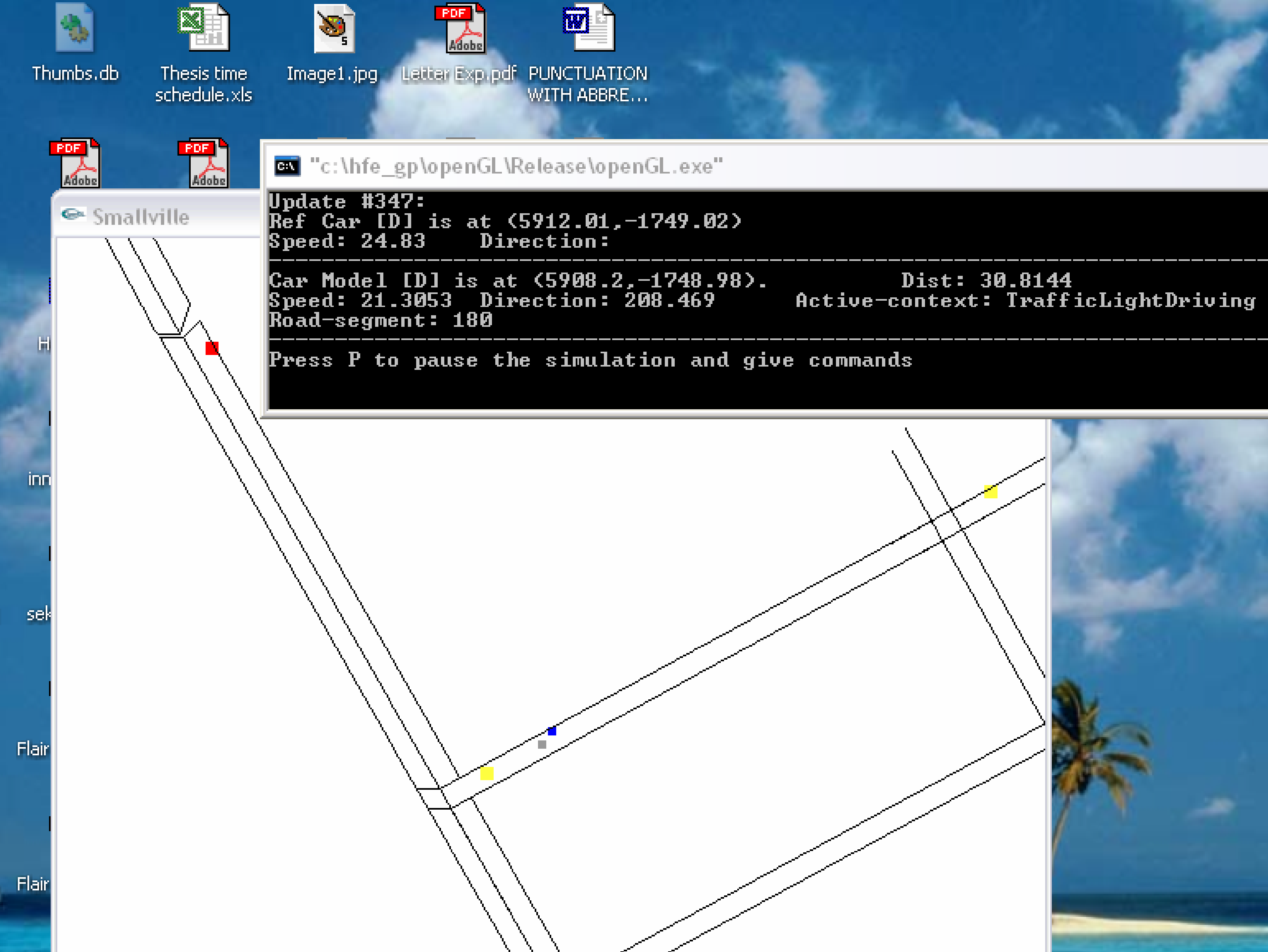
- Developed by Potter and DeJong, 1994.
- Makes it possible to have different populations evolving solutions to interdependent problems in parallel.
- The fitness function for individual in population 1 is not only a function of this individual, but also includes the best individual from population 2.
- Used primarily to evolve transition rules

Genetic Context Learning – GenCL



- GenCL Algorithm




Thumbs.db
Thesis time schedule.xls
Image1.jpg
Letter Exp.pdf
PUNCTUATION WITH ABBRE...
Smallville
"c:\hfe_gp\openGL\Release\openGL.exe"
Update #347:
Ref Car [D] is at (5912.01,-1749.02)
Speed: 24.83 Direction:
Car Model [D] is at (5908.2,-1748.98). Dist: 30.8144
Speed: 21.3053 Direction: 208.469 Active-context: TrafficLightDriving
Road-segment: 180
Press P to pause the simulation and give commands

Evaluation of GenCL

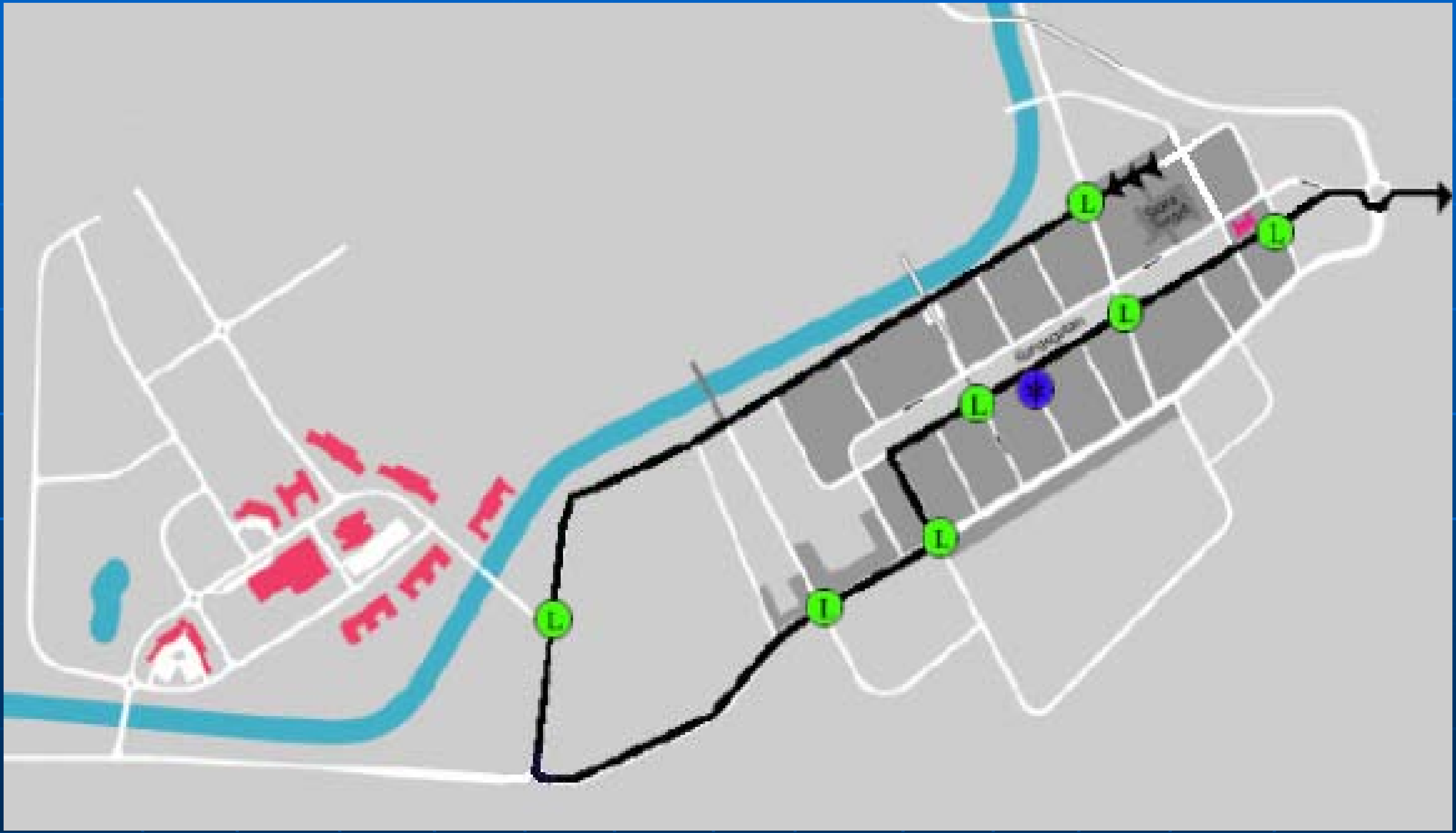
- Rigorously tested
- Objective:
  - Compare evolved agents with performance of corresponding test subject. (e.g., Agent A vs. Driver A; Agent B vs. Driver B, etc.)
  - NOT with optimal performance
  - Evaluate ability to generalize

Test Parameters

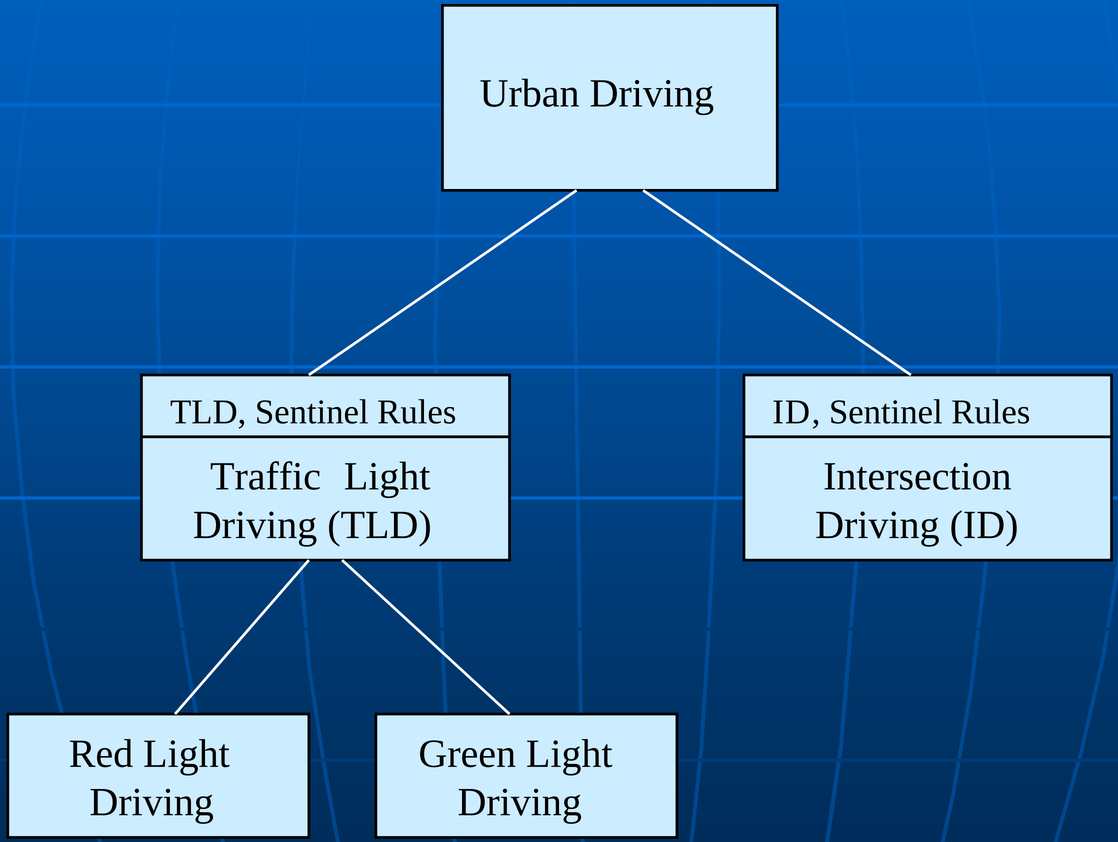
- Commercial Driving Simulator Used
- Five test subjects – students, male, 20-30 yrs old
  - Drivers A, B, C, D and E
- Two data sets:
  - Familiarization run – 15 min. to familiarize. Not recorded
  - Training – 20 minute run, A → B, through virtual city
  - Validation (4 mo. later) – 15 min, B → A, same city, diff. route (same test subjects as used in Training run)
- Urban driving only
  - Intersections, straight segments, traffic lights.
  - Realistic Environment - No repeated situations
  - Unpredictable behavior

Simulator

Virtual City



Pre-defined Context Hierarchy



Tests Performed

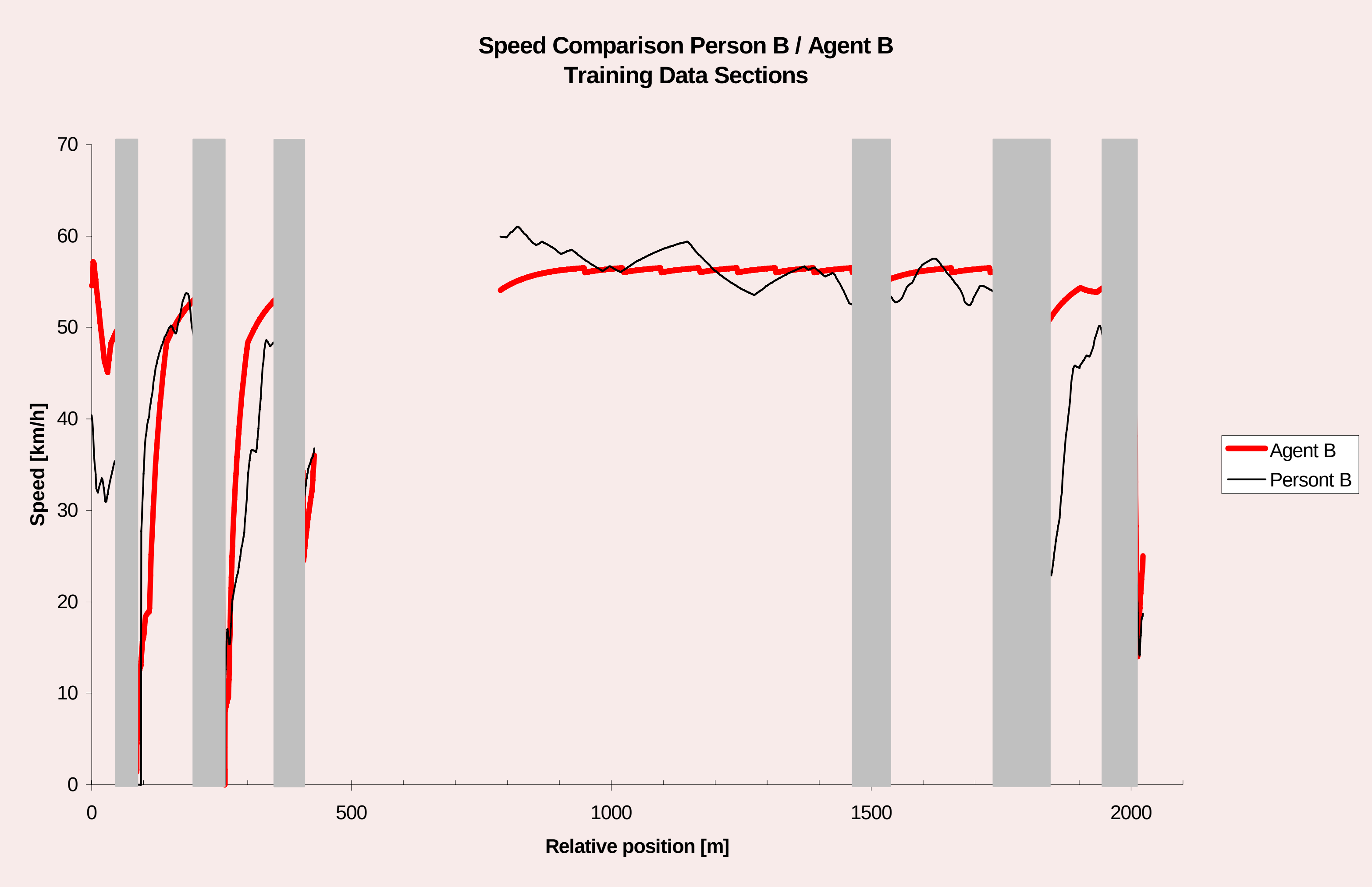
- Learning capability
  - Error rate on training data
- Generalization capability
  - Error rate on non-training data
    - Data from training run not used in training (“other”)
    - Data from validation run 4 months after training run
- Long term reliability
  - Measure of agent’s long-term stability
- Comparison with traditional techniques
  - Effectiveness compared to traditional techniques

Learning Capability

- Black-box testing of the training data

	Speed deviation [km/h]	Speed deviation %	Speed Correlation
Driver A/Agent A	1.92	3.14%	0.988
Driver B/Agent B	2.03	3.53%	0.983
Driver C/Agent C	1.85	3.41%	0.990
Driver D/Agent D	1.69	2.93%	0.989
Driver E/Agent E	3.81	6.25%	0.852

Generalization – Training Environment



Traffic Light Behavior

Training run

	Light 2	Light 3	Light 5	Light 6	Light 7	Light 9
Driver A	S	R*	S	R	R	S
Driver B	S	S	S	R	R	S
Driver C	S	S	S	S	S	S
Driver D	S	S	S	R	R	S
Driver E	R	S	S	R	R	S

Validation run

	Light 1	Light 2	Light 4	Light 5	Light 6	Light 7
Driver A	R	R	S	R	R	S
Driver B	R	R	S	R	R	R
Driver C	S	R	S	S	S	S
Driver D	R	S	S	R	S	S
Driver E	R	S	S	S	S	S

Generalization in Training Run

Qualitative validation - “other” training data

	Light 2	Light 3	Light 4	Light 6	Light 7	Light 8	Light 3b	Light 4b
Driver A/Agent A	S/S	R[1]/R	Ok	R/R	R/R	Ok	Ok	Ok
Driver B/Agent B	S/S	S/S	Ok	R/R	R/R	Ok	Ok	Ok
Driver C/Agent C	S/S	S/S	Ok	S/S	S/S	Ok	Ok	Ok
Driver D/Agent D	S/S	S/S	Ok	R/R	R/R	Ok	Ok	Ok
Driver E/Agent E	R/R	S/S	Ok	R/R	R/R	Ok	Ok	Ok

Quantitative validation - “other” tr. data

	Speed deviation [km/h]		Time deviation [s]		Speed
	RMS	Std.Dev.	RMS	Std.Dev.	Correlation
Agent A vs. Driver A	8.09	7.35	5.81	4.11	0.825
Agent B vs. Driver B	8.32	7.92	3.13	2.78	0.893
Agent C vs. Driver C	6.74	6.72	2.10	2.06	0.920
Agent D vs. Driver D	8.46	8.45	3.13	3.12	0.842
Agent E vs. Driver E	9.29	8.42	4.49	3.72	0.783

Generalization on Second Run

Qualitative validation

	Light 1	Light 3	Light 4	Light 5	Light 6	Light 7
Driver A/Agent A	R/R	OK	S/S	R/R	R/R	S/R
Driver B/Agent B	R/R	OK	S/S	R/R	R/R	R/R
Driver C/Agent C	S/S	OK	S/S	S/S	S/S	S/S
Driver D/Agent D	R/R	OK	S/S	R/R	S/R	S/R
Driver E/Agent E	R/R	OK	S/S	S/R	S/R	S/R

Quantitative validation

	Speed deviation [km/h]		Time deviation [s]		Speed
	RMS	Std.Dev.	RMS	Std.Dev.	Correlation
Agent A	7.47	7.44	1.47	1.47	0.880 (0.924)
Agent B	7.14	6.19	2.56	1.75	0.896
Agent C	7.12	7.11	3.60	2.80	0.926
Agent D	10.5	9.23	9.10	6.78	0.712 (0.860)
Agent E	17.0	12.0	38.4	30.3	0.550 (0.664)

Generalization

- Correlation between Agents and Drivers in the validation environment

	A	B	C	D	E
Agent A	0.879 (0.924)	0.840	0.831	0.708	0.667
Agent B	0.819	0.896	0.711	0.690	0.540
Agent C	0.853	0.644	0.926	0.857	0.913
Agent D	0.859	0.853	0.694	0.717 (0.860)	0.602
Agent E	0.794	0.855	0.738	0.675	0.550 (0.664)

The table is not symmetric since not the same data is used for row X / column Y as for row Y / column X.

Agents D and E

- Clearly Agents D and E were less successful in imitating their respective humans than A, B and C.
- Agent D confused the intersection with the traffic light
  - Came as result of insufficiently rich training data
- Agent E does not perform well because of the self-inconsistency of driver E

Long-term Reliability

- 40 minutes of simulation time, 70 traffic lights
  - Still running = intersection turning consistency

	Light turning Red			Light turning Green
	Stopping	Avg.Dist	Std.Dev	Correct behavior
Agent A	20/20	34.7	12.9	20/20
Agent B	22/22	8.04	1.95	22/22
Agent C	25/25	5.89	1.03	8/8
Agent D	31/34	4.50	1.31	6/6
Agent E	22/22	13.5	0.551	11/11

Usefulness

- Comparison to agent developed by Knowledge Engineer

Training environment

	Speed [km/h]		Time [s]		Speed
	RMS	Std.Dev.	RMS	Std.Dev.	Correlation
KE agent C vs. Driver C	7.94	7.81	4.35	4.35	0.894
GenCM agent C vs. Driver C	6.74	6.72	2.10	2.06	0.920
KE agent D vs. Driver D	8.83	8.88	9.55	9.01	0.852
GenCM agent D vs. Driver D	8.46	8.45	3.13	3.12	0.842

Validation environment

	Speed [km/h]		Time [s]		Speed
	RMS	Std.Dev.	RMS	Std.Dev.	Correlation
KE agent C vs. Driver C	8.52	8.38	4.05	3.10	0.902
GenCM agent C vs. Driver C	7.12	7.11	3.60	2.80	0.926
KE agent D vs. Driver D	9.02	8.64	7.43	7.21	0.876
GenCM agent D vs. Driver D	10.5	9.23	9.10	6.78	0.712

Ease of Use

- Non transforming algorithm
  - Able to use expert knowledge to tune the performance
- No pre-processing of the data
- Very small influence of GP settings
  - Individuals and Generations (Feldt & Nordin, 2000)

Conclusions and Results

GenCL features:

- Learns and generalizes well
- Reliable agents in long term
- Reflect individual behavior patterns
- Competitive with human modeling performance
- Learning in all context parts
- Can learn models from scratch, only requiring the predefinition of context hierarchy.

Disadvantages and Future Research

- A significant amount of manual data preparation is still necessary
  - Identify the contexts in the expert runs
  - Separate the contexts
  - Select the data for training from these contexts.
  - Run the GenCL algorithm manually
- On-going research to identify the contexts automatically – PhD dissertation by Mr. Viet Trinh