Traffic Light Control by Reinforcement Learning

Henrik Sejer Pedersen
Supervisor: Marco Chiarandini

Friday 16th November, 2018

Contents

1 Introduction

1.1 regular introduction stuff

1.2 Introduction to reinforcement learning

Reinforcement learning is a machine learning paradigm dedicated to solving sequential decision processes. The definition of a reinforcement learning algorithm given by Sutton and Barto [2018], is an algorithm which can solve a specific kind of sequential decision problem, namely those which can be described formally by a *finite Markov decision process*.

1.2.1 Finite Markov Decision Processes

The Markov decision process provides a formal description of sequential decision problems. In this project, we define a Markov decision process by a 4-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, p)$, where $\mathcal{S}$ is the state-space, $\mathcal{A}$ is the action-space, $\mathcal{R} \subset \mathbb{R}$ is the reward-space, and the function $p : \mathcal{S} \times \mathcal{R} \times \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ describes the conditional probability of entering a state $s' \in \mathcal{S}$ with reward $r \in \mathcal{R}$ given the previous state was $s \in \mathcal{S}$ and action $a \in \mathcal{A}$ was chosen. In finite Markov decision processes $\mathcal{S}$, $\mathcal{A}$ and $\mathcal{R}$ are all finite sets.

$$\sum_{s' \in \mathcal{S}} \sum_{r \in \mathcal{R}} p(s', r, s, a) = 1 \qquad \forall s \in \mathcal{S}, \forall a \in \mathcal{A}(s) \tag{1}$$

When solving Markov decision processes, a sequence of random variables is introduced. Let $T = \{1, 2, \ldots\}$ be a sequence of discrete time steps, S_t is then a random variable which indicates the state of the environment at time step t, A_t is the action chosen at time step t, from the set $\mathcal{A}(s)$ which denotes the actions available from state s. R_t is the reward received after choosing action A_{t-1} in state S_{t-1}

Figure 2 visualises the agent-environment loop. Initially, the agent in placed in some environment $\mathcal{E}$, from where the agent observes S_0. The agent the chooses some an action, yielding A_0, after the action has been performed, state S_{t+1} and R_t is presented to the agent, this results in a sequence of the form $S_0, A_0, R_1, S_1, A_1, R_2, S_2, \ldots$

$$p(s', r|s, a) \doteq \Pr\{S_t = s', R_t = r | S_{t-1} = s, A_{t-1} = a\}$$

1.2.2 RL

according to some policy π either *exploitatively* or *exploratively*.

1.3 Literature review

Abdulhai et al. [2003] Genders and Razavi [2018] Touhbi et al. [2017]

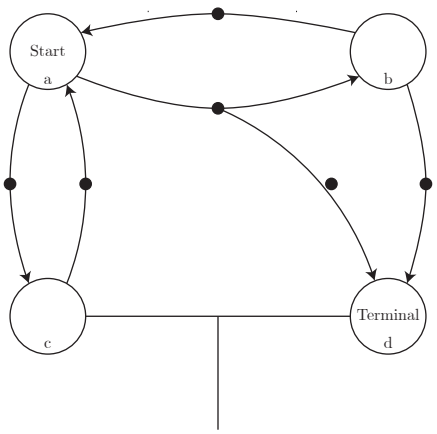


Figure 1: **TODO**

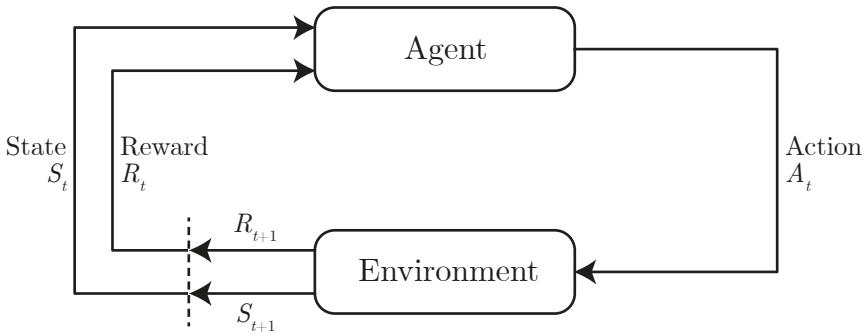


Figure 2: The Agent-Environment interface adapted from [Sutton and Barto, 2018]

Visual MDP representation for small example?

Figure 3: MDP represented as a graph

2 Instance definition

In this section, we introduce a traffic light located in southern Odense, Denmark and its implementation in the microscopic traffic simulator SUMO – *Simulation of Urban MObility*.

2.1 Network topology

The company SWARCO provided the data and network information we use in this project. We look at a traffic light controlled intersection located in southern Odense, Denmark, depicted in appendix A.1. We modeled the intersection in the microscopic traffic simulator SUMO, without the inclusion of bicycles, as the available data on bicycles is quite limited.

2.1.1 Induction Loops

In the network, a number of vehicle detectors are present, in the form of *induction loops*. In appendix A.1, the locations of induction loops relevant to the intersection is visualized. Figure 4 describes the meaning of the visualization.

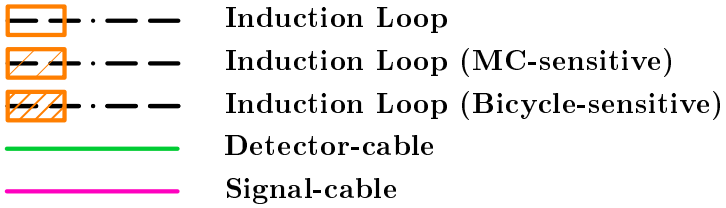


Figure 4: Induction loop signatures

The induction loops in the intersection cover vehicles quite well, but are quite lacking when it comes to bicycles, for this reason, bicycles are excluded from the project.

2.1.2 Traffic light movements

Appendix A.2 visualizes the traffic signal groups in the intersection. Table 1 describes the traffic signal strings on the drawing. A single signal can control multiple of these movements, denoted by, for instance, A1+A1v denotes a single signal controlling both left-turn and straight movements of A1.

string	meaning
A/B/C/...#	Upstream movement following road
A#Cy	Straight-ahead bicycle movement from A#
A#v	Left-turn movement from A#
A#h	right-turn movement from A#

Table 1: Explanation of traffic light movement strings for appendix A.2

todo: better string representation

2.2 SUMO Introduction

SUMO, short for *Simulation of Urban MObility*, is a microscopic, well-established general-purpose traffic simulator. It has been around since 2001 and is an open source project. What makes it particularly interesting for this project, is the ability to control elements of

the simulation externally, through a well-defined API. With this API, gathering information from the simulation for our agent is made simple, while also providing a way for our agent to control each traffic light.

2.3 Implementing the intersection in SUMO

SUMO uses a directed graph representation to define a traffic network, with some extra information. *Nodes* in the graph are points connected by one or more *edges*. Nodes contain connection data, which is (potentially) a many-to-many mapping of incoming lanes to outgoing lanes. The edges carry the lane information, allowing any number of adjacent lanes (within computational reason) to follow any single edge between two nodes. As such, each edge defines a set of up- or down-stream lanes, possibly consisting of multiple traffic movements.

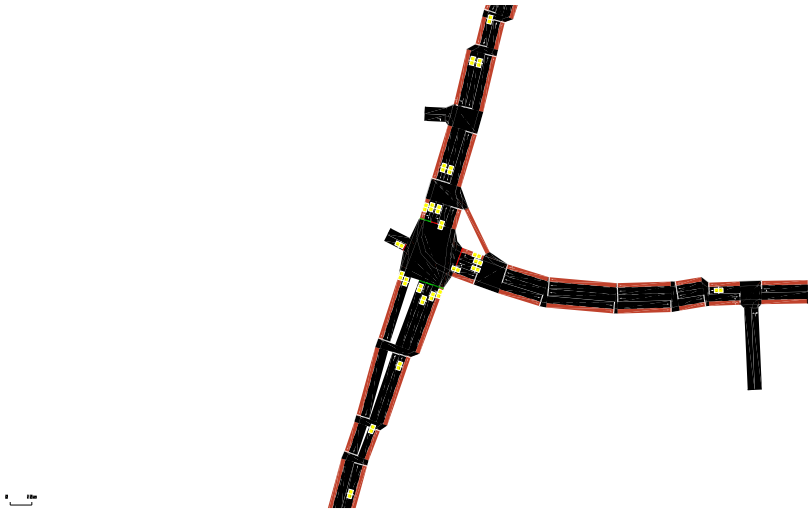


Figure 5: Intersection as implemented in SUMO, visualised by SUMO-GUI
todo: better image

The primary discrepancies between the actual intersection and the one implemented in SUMO are the missing induction loop D18. The induction loop is not present in the SUMO implementation as SUMO only allows placing induction loops on lanes.

2.4 Vehicle data

Along with technical drawings of the intersection, files describing the traffic flow exist in the form of 5 minutes aggregated readings from the 25 induction loops present in the intersection. To use this data in a meaningful way, we define orderings of the induction loops, each of which defines a route in the network.
Populating these sets with the input data can be modeled as an integer linear programming problem, and draws many parallels with the generalized set covering problem. Unfortunately, the induction loop readings have proven non-perfect, so we propose two different models.
The first model assumes that the induction loops never miss any vehicles, but may overcount. The second model assumes that the induction loops do not overcount, but may miss vehicles.

Both models give a vehicle count for each route for a single 5-minute interval and are called once for each such period. Solving for shorter intervals increase the network load in the simulation, we use the second model in this project.

2.4.1 Overcounts, no misses

Given a set of routes R, a set of detectors D, a binary route representation as defined below by B_{ij}, and upper bounds for *total* number of vehicles passing a detector, defined below by C_i. Select the number of vehicles to follow each route, such that the number of vehicles passing each detector is maximized for all detectors, given the single constraint:

- No more than C_i vehicles can pass detector D_i, $\forall i \in \{1, 2, \ldots, |D|\}$

For convenience we define the sets $I = \{1, 2, \ldots, |D|\}$, and $J = \{1, 2, \ldots, |R|\}$.
Let Bij be a binary constant such that:

$$B_{ij} = \begin{cases} 1 & \text{if route j passes detector i} \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in I, j \in J$$

Let C_i be an an detected number of vehicles at detector D_i, $\forall i \in I$ Let x_j be an integer variable with a lower bound of 0, indicating the number of vehicles following route $j, \forall j \; inJ$

$$\begin{aligned} \text{Maximize} \quad & \sum_{j \in J} x_j \cdot \sum_{i \in I} B_{ij} & & & \\ \text{s.t.} \quad & \sum_{j \in J} B_{ij} \cdot x_j & \leq & C_i & \forall i \in I \\ & x_j & \in & \mathbb{N} & \forall j \in J \\ & x_j & \geq & 0 & \forall j \in J \end{aligned}$$

The objective function maximizes the total sum of vehicles passing detectors. The outer sum sums over each route, and the second sum computes the number of detectors in the route from the outer sum.
The first constraint ensures that the sum of vehicles passing a detector does not surpass its capacity.
The second constraint ensures that the number of vehicles entering the simulation is integer.
The Third constraint ensures that the number of cars on each route must be non-negative.

2.4.2 Misses, no overcounts

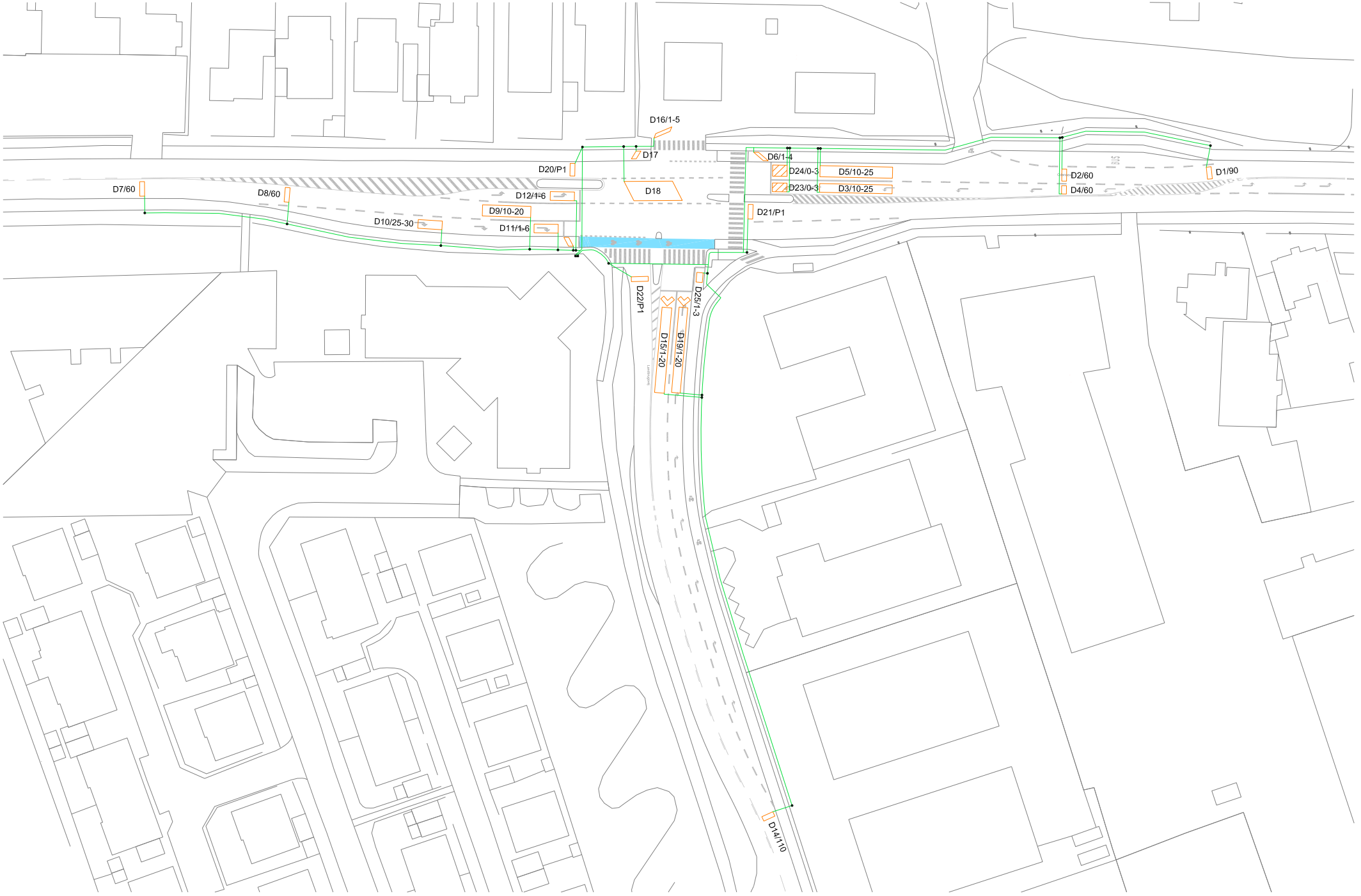
The previous model can with a few changes be modified, such that the vehicle counts act as lower bounds rather than upper bounds. Treating vehicle counts as lower bounds will put a more significant strain on the network, as more vehicles enter the simulation, which is desired.

$$\begin{aligned} \text{Minimize} \quad & \sum_{j \in J} x_j \cdot \sum_{i \in I} B_{ij} & & & \\ \text{s.t.} \quad & \sum_{j \in J} B_{ij} \cdot x_j & \geq & C_i & \forall i \in I \\ & x_j & \in & \mathbb{N} & \forall j \in J \\ & x_j & \geq & 0 & \forall j \in J \end{aligned}$$

The objective of the new model is to minimize the number of vehicles passing induction loops, while enforcing that at least the observed number of vehicles pass.
It is important to note that in this model, $B_{i\cdot}$ must contain at least *one* 1 to be feasible for all i, whereas this was not the case for the previous model.

A Intersection Technical Drawings

A.1 Technical drawing of intersection induction loops



0 2,5 5 7,5 10
1:500

swarco
SWARCO DANMARK A/S
Tonsbakken 16 - 18
DK 2740 Skovlunde
Telefon 36 88 88 88
Telefax 36 88 88 00

Odense Kommune
Odensevej - Landbrugsvej

Oversigtsplan

PROJ.	TEGN.	DATO
CJ	FJ	2017.0105

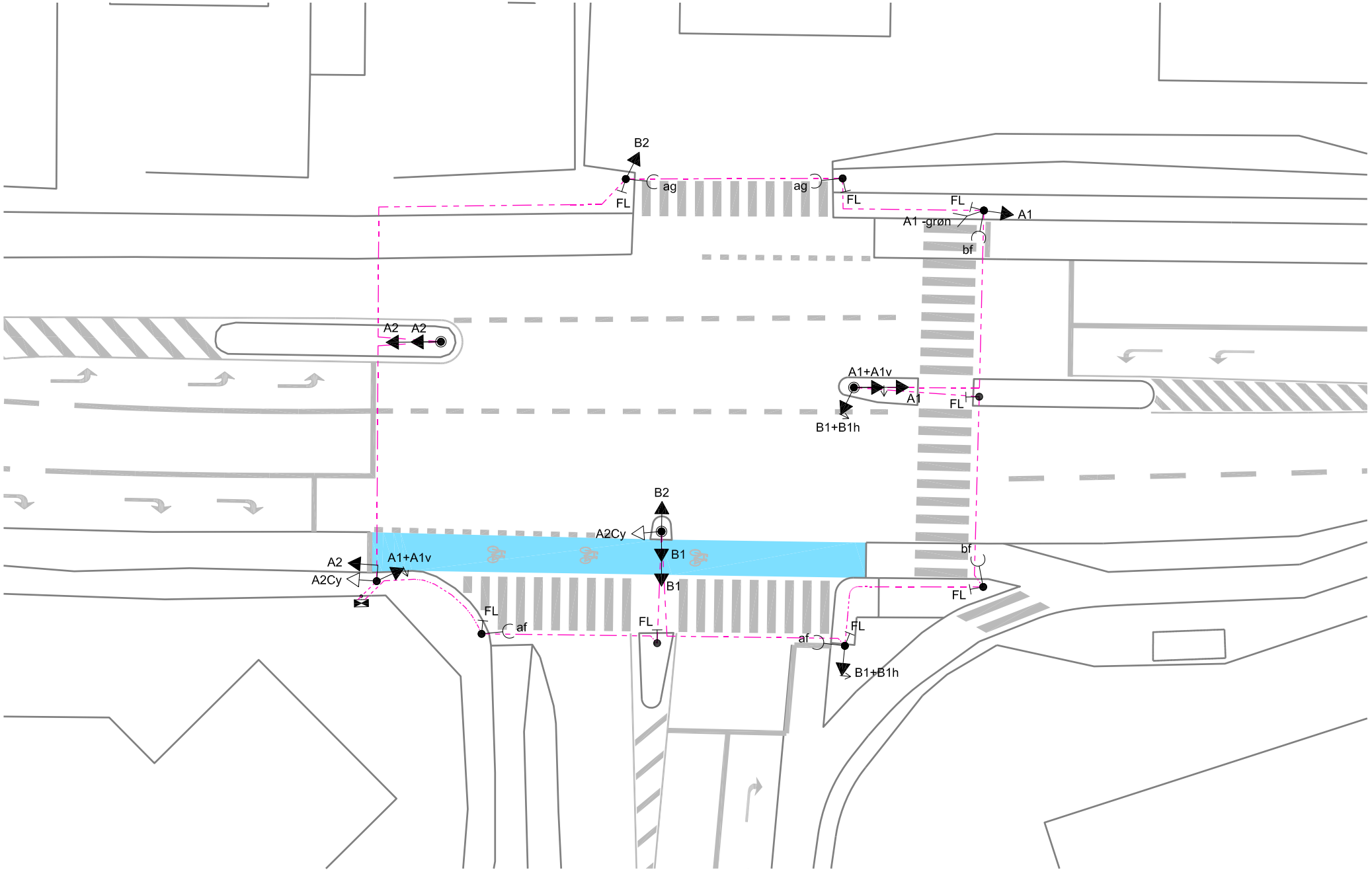
KUNDENR.

TEGNNR.	REV.
936,343	

Rev.12	Rev.11	Rev.10	Rev.9	Rev.8	Rev.7

Rev.6	Rev.5	Rev.4	Rev.3	Rev.2	Rev.1

A.2 Technical drawing of intersection movements



0 2.5 5 7.5 10

1:200

swarco
SWARCO DANMARK A/S
Tonsbakken 16 - 18
DK 2740 Skovlunde
Telefon 36 88 88 88
Telefax 36 88 88 00

Odense Kommune

Odensevej - Landbrugsvej

Oversigtsplan

PROJ.	TEGN.	DATO
CJ	FJ	2017.0105

KUNDENR.

TEGNNR.	REV.
936.343	

Rev.12 Rev.11 Rev.10 Rev.9 Rev.8 Rev.7 Rev.6 Rev.5 Rev.4 Rev.3 Rev.2 Rev.1

TODO: standardize references

References

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 2nd edition, 2018.

Baher Abdulhai, Rob Pringle, and Grigoris J. Karakoulas. Reinforcement learning for true adaptive traffic signal control. *Journal of Transportation Engineering*, 129(3):278, 2003. ISSN 0733947X.

Wade Genders and Saiedeh Razavi. Evaluating reinforcement learning state representations for adaptive traffic signal control. *Procedia Computer Science*, 130:26–33, 2018.

Saad Touhbi, Mohamed A. Babram, Tri Nguyen-Huu, Nicolas Marilleau, Moulay L. Hbid, Christophe Cambier, and Serge Stinckwich. Adaptive traffic signal control : Exploring reward definition for reinforcement learning. *Procedia Computer Science*, 109:513–520, 2017.

Kok-Lim Yau, Junaid Qadir, Hooi Khoo, Mee Ling, and Peter Komisarczuk. A survey on reinforcement learning models and algorithms for traffic signal control. *ACM Computing Surveys (CSUR)*, 50(3):1–38, 2017;2018;.

Kok-Lim Alvin Yau, Junaid Qadir, Hooi Ling Khoo, Mee Hong Ling, and Peter Komisarczuk. A survey on reinforcement learning models and algorithms for traffic signal control. *ACM Comput. Surv.*, 50(3):34:1–34:38, June 2017. ISSN 0360-0300. doi: 10.1145/3068287. URL http://doi.acm.org/10.1145/3068287.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *CoRR*, abs/1602.01783, 2016. URL http://arxiv.org/abs/1602.01783.