1 Introduction

Controllers can be designed by machine learning using different kinds of information available to the learning system. Approaches like reinforcement learning, genetic algorithms and neural networks typically don't use prior knowledge about the system to be controlled. This results in time-consuming experimentation with the dynamic system, low success rate of learning, low robustness and lack of interpretability of learned controllers. Humans, however, rarely attempt to learn from scratch. They extract initial biases as well as strategies from their prior knowledge of the system or from demonstration of experienced operators.

The idea of *behavioural cloning* (a term introduced by Donald Michie [10]), but probably first carried out by Donaldson [6]) is to make use of the operator's skill in the development of an automatic controller. A skilled operator's control traces are used as examples for machine learning to reconstruct the underlying control strategy that the operator executes subconsciously. The goal of behavioural cloning is not only to induce a successful controller, but also to achieve better understanding of the human operators subconscious skill [14]. The first such rule-based acquisition of real-time control skill was in [4]. Behavioural cloning was later successfully used in problem domains as pole balancing, production line scheduling, piloting [12] and operating cranes. These experiments are reviewed in [2]. Controllers were usually induced in the form of decision or regression trees.

Although such clones do provide some insight into the control strategy, they in general lack conceptual structure that would clearly capture the causal relations in the domain and the goal structure of the control strategy. Clones in the form of decision trees do not explicitly show any time ordering between events in the controlled system and the actions taken. Ignoring causality and the dynamics of the system usually results in low robustness of induced controller. Logic-based approaches to machine learning, in particular Inductive Logic Programming (ILP) [11], enable the use of background knowledge, that is knowledge known prior to learning. Specially because of their ability to flexibly accommodate background knowledge, ILP techniques would seem to be more appropriate. Some experiments with ILP were performed in the piloting domain [3], but the results were generally not better than those with decision or regression trees.

Although successful clones have been induced in the form of trees or rule sets, the following problems have generally been observed with this approach:

- Typically, induced clones are brittle with respect to small changes in the control task.
- The clone induction process typically has low yield: the proportion of successful controllers among all the induced clones is low, typically well below 50%.
- Resulting clones are purely reactive and inadequately structured as conceptualizations of the human skill. They lack typical elements of human control strategies such as goals, subgoals, phases and causality.

Another approach proposed in [16] exploits some results from control theory. This approach considers the system's dynamics and practically enables identifying discrete subgoals. It improves both the clones' robustness with respect to changes in the control task, and the yield of the cloning process. However, this approach still has difficulties in domains with significant

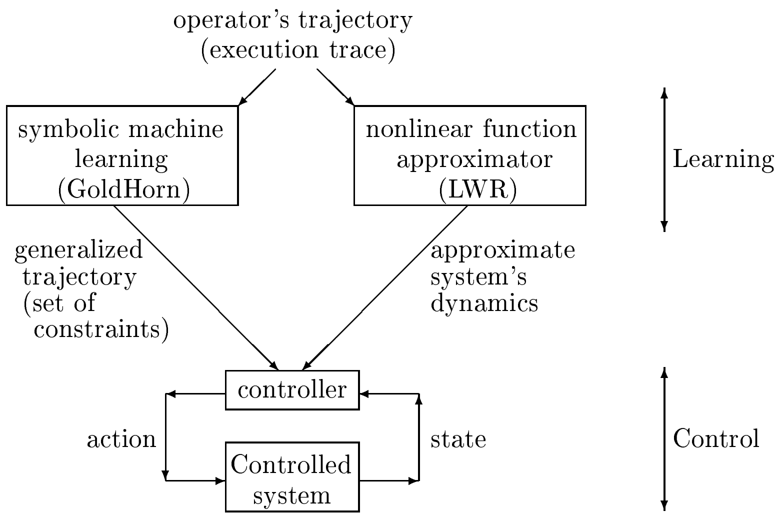

Figure 1: Reconstructing human control skill from operator's trajectory.

nonlinearities or where the operator's control plan is not simply expressed in terms of a few discrete subgoals.

In this paper we propose a new approach to behavioural cloning, suitable also for strongly nonlinear domains. The trajectory the operator is trying to follow is generalized separately from the system's dynamics and can be viewed as a *continuous subgoal.* In particular, we do not learn the trajectory in time, but rather the constraints among the state variables in the execution trace. These constraints determine the corresponding desired trajectory to the goal, also for system states other than those explicitly mentioned in the operator's execution trace. Actions that maintain the desired trajectory are computed using knowledge of the system's dynamics, learned by nonlinear function approximators. The experiments performed in the crane domain in this paper, and previous experiments in the Acrobot domain [17] demonstrated that this approach significantly improves the yield of the cloning process and provides a good insight in the operator's subcognitive skill. Qualitative strategy, generalized from the operator's trajectory, is comprehensible and offers the possibility to improve the operator's control strategy.

The structure of the paper is as follows. Section 2 gives a general description of our approach. The following two sections present the domain of container cranes and experiments where we used our approach to reconstruct the human crane control skill. In section 5 we generalize the induced strategy into a comprehensible qualitative strategy. By transforming this qualitative strategy into operational quantitative strategies in section 6, we show that the qualitative strategy is general and successful. In section 7 we test the robustness of the resulting controllers. Finally, we discuss some points of interest and give conclusions.

2 Our approach to skill modelling

2.1 General idea

The main idea of our approach (see Fig. 1) is to generalize the trajectory the operator is trying to follow and separately learn the system's dynamics by a nonlinear function approximator. The learned nonlinear model of system's dynamics is then used to compute control action which achieves the desired next state on the trajectory. When generalizing a given operator's trajectory, we do not learn the trajectory in time, but rather induce constraints between the state variables in the operator's execution trace (or traces). Usually, the induced constraints involve only some of the state variables and have the form of differential equations.

These constraints can be used to control the system as follows. Ideally, to mimic the operator's control strategy, the constraints should hold in every state. So an appropriate control action should minimize the deviation of the predicted next state from the constraints. A measure of such deviation of a state from the constraints will be called *constraints error*. A constraints error simply measures the degree to which a system's state does not satisfy the constraints. Given a system state and the constraints, an appropriate action can be computed in many ways. In general it requires a model of the system's dynamics and minimization of the constraints error over the possible actions. If it is possible to express one state variable as a function of other state variables in the constraints, and this state variable is directly influenced by control action, the minimization of error in this state variable determines the action to take. Another possibility, used also in this paper, is to induce constraints between the current and the next state in the execution trace and choose the action which minimizes error w.r.t. constraints in the next state.

One motivation for this approach to skill modelling is in the possibility of explaining the operator's control strategy. Induced symbolic constraints that characterize the operator's strategy can be used to explain such a subcognitive strategy. As shown later in the paper, induced constraints can also be turned into qualitative control rules, which can give additional insight into the operator's control strategy and offers a possibility to optimise the operator's strategy.

2.2 Using GoldHorn to generalize the operator's trajectory

The operator's trajectory is generalized by finding constraints among the state variables. To induce such constraints in the form of ordinary differential equations, we used the machine learning program called GoldHorn [9]. Given the behavior of the system, i.e. a sampled execution trace, GoldHorn attempts to find a set of differential equations that describe the dynamics of the system. GoldHorn does not simply fit the parameters of equations of given forms, but it also constructs new forms of equations. To do this, GoldHorn first introduces new terms by repeatedly applying operators, such as multiplication, to the state variables and their time derivatives. Then, given the set of all the terms, including the original variables and the newly constructed terms, differential equations are generated from these terms using linear regression. The significance of the equation is judged by two measures: the multiple correlation coefficient and the normalized deviation. It should be noted that as a result, GoldHorn not only finds differential equations of some predefined forms, but also does *structural* synthesis of new forms of equations.

GoldHorn is an extension of the program Lagrange [7]. The improvements with respect to Lagrange effectively deal with noisy data. This makes GoldHorn applicable to the modeling of real world systems from imperfect measured data.

The generalized operator's trajectory is induced by GoldHorn from one or more execution traces. Control actions are deliberately left out from the traces, since they can be computed if we know the desired next state and have an approximate model of the system. Similarly, we can leave out some of the state variables (typically the time derivatives of other state variables), to keep the induced constraints as simple as possible. GoldHorn then induces constraints among the remaining variables in the form of differential equations and ranks them according to their significance (error estimates). One or more of the most significant induced equations are used as constraints which define the generalized operator's trajectory.

Of course, it would be possible in principle to learn control actions directly from example traces. However, experiments show that inducing generalized trajectories leads to much more robust controllers.

2.3 Learning the system's dynamics

The system's dynamics can be learned by any nonlinear function approximator from some execution traces. We use locally weighted regression [5, 13], since it enables incremental learning and provides local linear model of the system's dynamics near the current state. Locally weighted regression is a kind of memory based learning, so no generalization is actually done in the training phase. We just need to store the observed points in the state space. In the prediction phase, points in the state space are weighted according to the distance from the query point and locally weighted linear model is computed. After answering the query the local model is discarded and a new locally weighted model is created to answer the next query. This provides an easy and efficient way of incremental learning with no negative interference. When the system is controlled, each current state is simply stored, and this experience can be used to predict the system's behavior in nearby points in the state space. If we expect the system's dynamics to change in time, only recent experience is considered in the prediction phase.

3 Container crane

In this section we describe experiments in the domain of container cranes. This domain was earlier used in [14].

To transport a container (see Fig. 2) from the shore to a target position on the ship, two operations are to be performed: positioning of the trolley, bringing it above the target load position (X_g), and rope operation, bringing the load to the desired height (L_g). The performance requirements include basic safety, stop-gap accuracy and as high capacity as possible. The last requirement means that the time for transportation is to be minimized. Consequently, the two operations are to be performed simultaneously. The most difficult aspect of the task is to control the swing of the rope. When the load is close to the goal position, the swing should ideally be zero.

A crane simulator was used in our experiments. The parameters of the system (lengths, heights, masses, etc.) are the same as those of the real cranes in Port of Koper. The state of the system is specified by six variables:

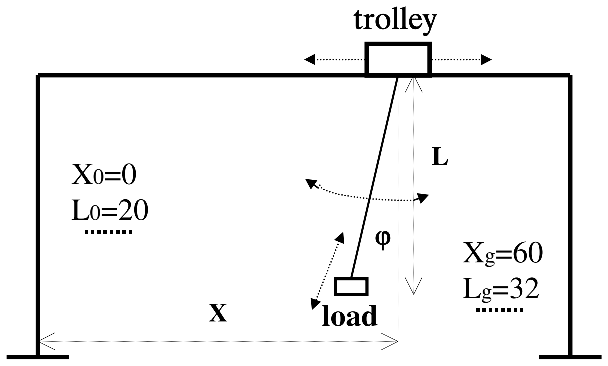

Figure 2: Container crane: the state of the system is specified by six variables: trolley position X and its velocity $\dot{X}$, rope inclination angle ϕ and its angular velocity $\dot{\phi}$, rope length L and its velocity $\dot{L}$. The system is controlled through force to the trolley in the horizontal direction and force in the direction of the rope. The task is to transport the load from its start position ($X_0 = \dot{X}_0 = \phi_0 = \dot{\phi}_0 = 0$, $L_0 = 20$, $\dot{L}_0 = 0$) to the goal position ($X_g = 60, \dot{X}_g = \phi_g = \dot{\phi}_g = 0$, $L_g = 32$, $\dot{L}_g = 0$).

trolley position X and its velocity $\dot{X}$, rope inclination angle ϕ and its angular velocity $\dot{\phi}$, rope length L and its velocity $\dot{L}$. Two control forces are applied to the system: force to the trolley in the horizontal direction and force in the direction of the rope.

We used experimental data from manually controlling the crane from a previous study [14]. In that study, six students volunteered to learn to control the simulator. All of them succeeded to accomplish the task. However, remarkable individual differences were observed regarding the speed of controlling and the characteristics of the strategy they used. Some operators tended towards fast and less reliable operation, others were more conservative and slower, in order to avoid large rope oscillations.

Previous experiments in this domain involve behavioural cloning with regression trees [14], combining skills from several operators [15] and experiments with LQR clones with subgoals [16]. The main problem for regression trees was the swing control, i.e. large rope oscillations when the trolley approached its goal position. For this reason slower, more conservative traces were found by far the most useful for cloning with regression trees. On the other hand, strategies of faster operators are more complex, requiring the exact timing and skill to decrease large rope oscillations due to large trolley accelerations. In this paper we use traces of the fastest operator, that is the operator who achieved shortest times. He was able to afford large initial swing caused by large acceleration, and later skillfully reduce the swing.

4 Skill reconstruction in the crane domain

4.1 Reconstructing the human crane control skill

Reconstruction of human control skill generally follows the approach described in section 2. Here we give details specific to the crane domain.

First the generalized trajectory the operator is trying to follow is induced from one or more traces of the same subject. In the case of the crane control, two forces are applied to the system: the force to the trolley XF and the force to the rope YF. So the generalized trajectory consists of two variables: the desired trolley velocity DX_{des} and the desired rope length L_{des}. Of course, generalized trajectory could be expressed in a different way, for example (DX_{des},DL_{des}). We decided for (DX_{des},L_{des}) trajectory since it is easy to understand. The learned trajectory consists of DX_{des} and L_{des} as a function of other state variables, that is DX and L in the next state as the function of the current state. In fact, DX_{des} and L_{des} are learned separately, but they form one single trajectory since they are functions of the same variables. It is important to note that we do not learn the (DX_{des},L_{des}) trajectory in time, but rather constraints between the state variables in the form of differential equations induced by the GoldHorn program from the execution trace. Most significant equations are used to express the desired DX_{des} and L_{des} as a function of the current state.

The system's dynamics is learned using locally weighted regression from some execution traces, as described in section 2. In the case of crane control we are interested in the change of DX (L) as a linear function of other state variables and force XF (YF) near the current state. The desired force XF (YF) is expressed from this linear function. If we are able to compute the desired DX and L as a function of the state, have an approximation of the system's dynamics, and we know the time step dt after which the desired DX and L has to be achieved, than an action pair (XF,YF), which moves towards the desired position can be computed.

When the trajectory is known, the action which achieves the desired next state on the trajectory can be computed using a locally linear model. So, the actual controller consists of two parts: nonlinear model of system's dynamics (which is updated online) and a desired trajectory. At each time step the controller considers the current state, estimates local linear model around the current state and computes action pair (XF,YF) which makes one step towards the desired next state on the trajectory.

4.2 Notes on experiments

We used traces sampled at sample time $dt = 0.2$ sec. DX_{des} and L_{des} were computed as $DX_{des}(t) = DX(t + dt)$ and $L_{des}(t) = L(t + dt)$. To learn the desired trajectory of the trolley velocity DX_{des}, GoldHorn was used to learn the constraints in the trace among variables DX_{des}, X, ϕ and $\dot{\phi}$. The trajectory of rope length L_{des} was learned similarly. To learn the system's dynamics with locally weighted regression, the original trace and another two traces of the same operator were used.

4.3 Learning quantitative strategy

This approach to skill reconstruction turned out to be generally very effective. For every successful execution trace that we experimented with, GoldHorn induced a generalized trajectory which allowed the controller to complete the task. The induced control strategy often concludes the task faster than the original operator's traces. However, the time needed to complete the task varied significantly with example execution trace and the particular differential equation used. We found that differential equations with linear and quadratic factors were usually most successful. Terms of higher orders usually did not improve the performance. Another interesting, but not surprising observation was, that usually a more successful trajectory was induced from a faster execution trace. So individual differ-

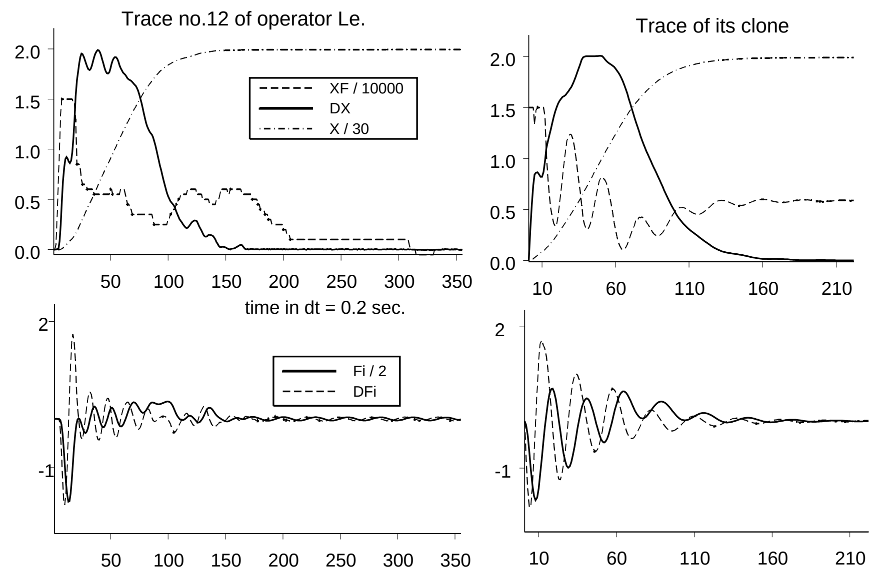

Figure 3: Original and its clone: on the left is the original trace and on the right is the trace of its clone, using the induced quantitative strategy given in equations 2 and 3. The original concludes the task in 72 sec. The clone is much faster and concludes the task in 44 sec. Decreasing of the rope swing can be observed in both traces.

ences between the operators, like differences in operator's speed and skill to decrease the swing of the rope, were reflected in the generalized trajectories induced from their traces.

In the next paragraphs we present the strategy learned from one of the traces of the fastest operator (see Fig. 3). To generalize the operator's (DX_{des}, L_{des}) trajectory GoldHorn was used to learn the constraints in the trace among the state variables as differential equations with linear or quadratic terms. Since controlling the trolley is considerably harder than controlling the rope, we focus on DX_{des} trajectory.

The desired trajectory of the trolley velocity DX_{des} (eq. 2) was expressed from the most significant equation (eq. 1), induced from the operator's trace:

$$509.52 + 51.24X + 28.21\phi - 564.32DX_{des} - X^2 = 0 \tag{1}$$

$$DX_{des} = 0.902 - 0.0018X^2 + 0.090X + 0.050\phi \tag{2}$$

To control the rope, a very simple equation with higher error estimate was used:

$$L_{des} = 0.0037X^2 + 17.46$$

The equation in simple, but not accurate enough, since it states that L_{des} at the goal position X_g is 30.7. To conclude the task, the rope length at the goal position should be $L_{des}(X_g)$=L_g=32. This can be corrected by adding this difference to the learned L_{des}, as it is discussed in section 8. In this way we get the rule to control the rope length:

$$L_{des} = 0.0037X^2 + 17.46 + 1.3 \tag{3}$$

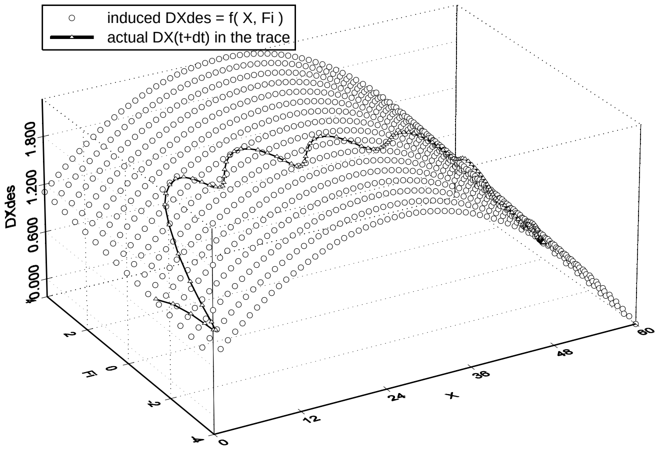

Figure 4: Induced quantitative strategy: the dotted surface represents the induced quantitative strategy (eq. 2) DX_{des} as a function of X and ϕ. The line represent the actual $DX(t+dt)$ from the original operator's trace. Qualitative rules (5) $Q_+(DX_{des}, \phi)$ and the rule *if* $X < 25.62$ *then* $Q_+(DX_{des}, X)$ *else* $Q_-(DX_{des}, X)$ can be observed.

Although this rule is simple, it is very successful, robust any easy to understand: the operator brings down the load as the trolley moves from its start to the goal position.

The rules given by equations 2 and 3 describe the generalized operator's trajectory. The actual controller consists of the generalized trajectory and the learned system's dynamics as described in section 2. The controller was tested on the simulator. It concludes the task in 44 sec. and is much faster than the original operator's trace, which needs 72 sec. Both, the operator and the clone (see Fig. 3), stop the trolley before the exact goal position ($X = 59.8$) of the trolley ($X_g = 60$), but are successful since the trolley is in its tolerance area. A crucial part of the crane control strategy, that is decreasing of the rope swing, can be observed in both traces. This is done by acceleration of the trolley when the rope angle increases, and can also be observed in the induced DX_{des} on Fig. 4.

5 Qualitative strategy

In this section we generalize the induced quantitative strategy into a qualitative strategy. We show that this qualitative strategy is easy to understand (actually it gives all the qualitative knowledge needed to conclude the crane task), general (it is not sensitive to the exact settings of the qualitative parameters) and very successful.

Since controlling the trolley is considerably harder than controlling the rope, we focus on DX_{des} trajectory. In all experiments we used L_{des} trajectory given by rule 3. Of course, the same approach as with DX_{des} trajectory could be used to generalize L_{des} trajectory into a qualitative strategy.

One way to express a qualitative strategy is by two qualitative proportionality predicates Q_+ and Q_- as defined by Forbus [8]. The notation $Q_+(y, x)$ means that y is a function of a set of variables, including x, so that y is monotonic increasing (strictly increasing) in its dependence on x:

$$Q_+(y,x) \Leftrightarrow \frac{\partial y(x,...)}{\partial x} > 0 \tag{4}$$

We say that y is *positively related* to x. The representation $Q_+(y,x)$ allows to conjecture that the qualitative behavior of y cannot be controlled without taking into account the qualitative behavior of x and furthermore, positive (or negative) changes in x will be manifested in changes of y that are more positive (or negative) than they would have been without the change of x. The meaning of $Q_-(y,x)$ is analogous but in the opposite direction.

Note that this is a weaker relation than the other well known qualitative relation $M_+(y,x)$. $Q_+(y,x)$ only states that when x rises, y will also, barring other changes. For example, we could express the fact that the pressure of gas in a container increases as the temperature increases by $Q_+(Pressure, Temperature)$ even if the quantitative equation is $Pressure \times Volume/Temperature = const$. It can happen that the pressure decreases even if the temperature increases because of a change in the volume; however we can not write $M_+(Pressure, Temperature)$.

Qualitative constraints defining a qualitative strategy can now be derived by qualitative abstraction of the quantitative strategy given by eq. 2:

$$\begin{aligned}
\frac{\partial DX_{des}}{\partial \phi} =& 0.050 > 0 \\
& \Rightarrow Q_+(DX_{des}, \phi) \\
\frac{\partial DX_{des}}{\partial X} =& -2X + 51.24 \\
& \Rightarrow Q_+(DX_{des}, X) \text{ if } X < 25.62 \\
& \quad Q_-(DX_{des}, X) \text{ if } X > 25.62
\end{aligned}$$

These three rules describe the qualitative control strategy:

$$\begin{array}{c}
Q_+(DX_{des}, \phi) \\
\text{if } X < X_{mid} \text{ then} \\
Q_+(DX_{des}, X) \\
\text{else } Q_-(DX_{des}, X)
\end{array} \tag{5}$$

One form of qualitative rule for $DX_{des}(X, \phi)$, consistent with 5, is:

$$\begin{array}{c}
\text{if } X < X_{mid} \text{ then} \\
DX_{des} = M_x^+(X) + M_{\phi 1}^+(\phi) \\
\text{else } DX_{des} = M_x^-(X) + M_{\phi 2}^+(\phi)
\end{array} \tag{6}$$

Note that parameters X_{mid}, M_x^+, M_x^-, $M_{\phi 1}^+$ and $M_{\phi 2}^+$ are *qualitative*. M^+ (M^-) are arbitrary strictly increasing (decreasing) functions for $X < X_{mid}$ ($X > X_{mid}$).

Qualitative rule $Q_+(DX_{des}, \phi)$ is very important part of the operator's strategy and describes his skill to decrease the swing of the rope. By acceleration of the trolley when the rope angle increases, the angular velocity is decreased. The other two qualitative rules just say that the operator first increases the trolley's velocity (when the trolley is near its start position: $X < X_{mid}$) and decreases it later.

The induced qualitative strategy, given by rule 6, defines a set of quantitative strategies S with free parameters X_{mid}, M_x^+, M_x^-,$M_{\phi 1}^+$ and $M_{\phi 2}^+$. To transform the qualitative strategy into an operational quantitative strategy, the qualitative parameters have to be concretised into quantitative values or functions. For example, the qualitative parameters of the quantitative strategy of eq. 2 are:

$$X_{mid} = 25.62$$
$$M_{\phi 1}^+(\phi) = M_{\phi 2}^+(\phi) = 0.050\phi$$
$$M_x^+(X) = M_x^-(X) = 0.902 - 0.0018X^2 + 0.090X$$

6 Transforming qualitative strategy into operational quantitative strategies

The induced qualitative strategy, given by rule 6, defines a set of quantitative strategies S with free qualitative parameters X_{mid}, M_x^+, M_x^-, $M_{\phi 1}^+$ and $M_{\phi 2}^+$. Clearly, some of those strategies are not successful (for example if $M_x^+(0) < 0$). Here we investigate how successful and general is the learned qualitative strategy, that is how sensitive it is to changes of the qualitative parameters. In order to do this we performed two sets of experiments:

- In the first experiment we used just basic knowledge of the control task (like goal position and allowed ranges of the state variables) and randomly generated functions satisfying the qualitative constraints. Results show that the qualitative strategy is general and that the exact selection of the qualitative parameters is not crucial for its success.
- In the second experiment we restricted qualitative functions to simple (linear and quadratic) functions, and used knowledge of the control task to find constraints among parameters of the quantitative strategy. Experimentally it is confirmed that, by using some knowledge of the control task, it is easy to find the constraints among parameters of the quantitative strategy which filter out bad strategies. In this way, very successful strategies can be obtained. Since a generalized trajectory is expressed in a symbolic way, it is easy to understand, analyze and correct.

6.1 Randomly generated qualitative functions

Here we use basic knowledge of the control task and randomly generated functions with the required qualitative properties, to transform the induced qualitative strategy into operational quantitative strategies. Knowledge used consists of the task limits ($|\phi| < \phi_{max}$, $|DX_{des}| < DX_{max}$) and two simple facts of the control task:

1. the trolley starts towards the goal: $DX_{des}(0,0) \in (0, DX_{max})$
2. the trolley stops at X_g: $DX_{des}(X_g, 0)$=0

First we define a set of strictly increasing (decreasing) functions and use them to incorporate the knowledge of the control task into a set of quantitative strategies, based on induced qualitative strategy. Then experimental results are presented.

Let us define a set of functions F^+ (F^-), such that any $f^+ \in F^+$ ($f^- \in F^-$) is strictly increasing (decreasing) function defined on $[0,1]$ and mapping to $[0,1]$:

$$\begin{aligned} &f^+, f^- : [0,1] \to [0,1] \\ &f^+(x_1) > f^+(x_2) \Leftrightarrow x_1 > x_2 \\ &f^-(x_1) < f^-(x_2) \Leftrightarrow x_1 > x_2 \end{aligned}$$

Similarly we define a set of functions H^+ (H^-), such that any $h^+ \in H^+$ ($h^- \in H^-$) is strictly increasing (decreasing) function defined on $[-1,1]$ and mapping to $[-1,1]$, with the property that $h^+(0) = h^-(0) = 0$.

Different functions f^+ can be generated by taking $R+1$ random numbers sorted in increasing order $0 \le y_i \le 1$, $i = 0,1,\ldots,R$ ($y_i < y_{i+1}$) and set $f^+(i/R)$ to y_i. The values at other points are linearly interpolated: $f^+(x) = y_j + (y_{j+1} - y_j)(Rx - j)$ where $j = \lfloor xR \rfloor$. Functions h^+ can be generated similarly as f^+ functions, by taking $R/2$ random numbers from $[-1,0)$, number 0, and R / 2 random numbers from $(0,1]$ and sort them in increasing order.

The basic knowledge of the control task can be incorporated into a set of quantitative strategies based on qualitative strategy (rule 6) as follows. Since $|\phi| < \phi_{max}$, increasing functions $M^+_{\phi 1}$ and $M^+_{\phi 2}$ can be written as:

$$M^+_{\phi 1,\phi 2}(\phi) = k_1 h^+_{1,2}(\frac{\phi}{\phi_{max}})$$

Since $|DX_{des}| < DX_{max}$, $k_1 \in (0, DX_{max})$. Because the trolley must start towards goal ($DX_{des}(0,0) > 0$) and $|DX_{des}| < DX_{max}$ and $M^+_x(X)$ is increasing function, M^+_x can be written as:

$$M^+_x(X) = DX_{max} f^+_1(X/X_{mid})$$

Since the trolley must stop at the goal position ($DX_{des}(X_g,0)$=0) $M^-_x(X)$ can be written as:

$$M^-_x(X) = DX_{max} h^-_3(\frac{X - X_g}{X_g - X_{mid}})$$

This gives a set of quantitative strategies $S_{q0} \subset S$:

$$\begin{aligned} &\text{if } X < X_{mid} \text{ then} \\ &\quad DX_{des} = DX_{max} f^+_1(\frac{X}{X_{mid}}) + k_1 h^+_1(\frac{\phi}{\phi_{max}}) \\ &\text{else} \\ &\quad DX_{des} = DX_{max} h^-_3(\frac{X - X_g}{X_g - X_{mid}}) + k_1 h^+_2(\frac{\phi}{\phi_{max}}) \\ &X_{mid} \in (0, X_g),\ k_1 \in (0, DX_{max}) \end{aligned} \tag{7}$$

These quantitative strategies were tested on the simulator. X_{mid} was set to values 15, 30, 45 and k_1 to values $DX_{max}/2$, $DX_{max}/4$ and $DX_{max}/8$. For each of these parameter settings, we generated 10 sets of random functions f^+_1, h^+_1, h^+_2, h^-_3 with $R = 10$. In this way we get a set of $3 \times 3 \times 10 = 90$ different quantitative strategies referred to as S_{q0}. Considering the fact that we used a qualitative strategy, where qualitative functions were random and we used only the basic knowledge of the crane task, results are very good. Out of 90 different quantitative strategies 75 finished the task in 180 sec. or less. 39 strategies completed the task in less than 80 sec., some of them faster than 50 seconds. The details are given on Fig. 5.

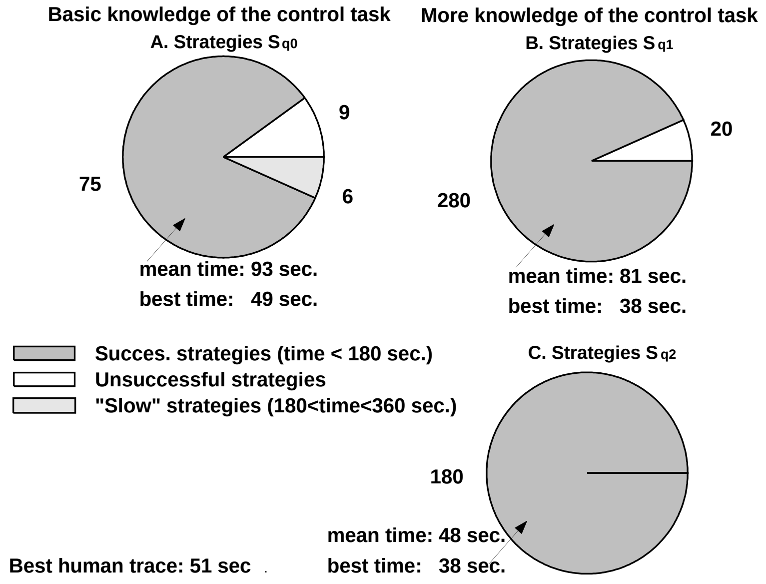

Figure 5: Efficiency of the qualitative strategy: Pie A. gives results with randomly generated qualitative functions. Out of 90 different quantitative strategies 75 finish the task in 180 sec. or less. Another 6 are successful in 360 sec. Pies B. and C. give results with qualitative functions using knowledge of the control task. All of 180 strategies from S_{q2} are successful and very fast. Even their mean time (48 sec.) is better than the time of the fastest human trace (51 sec.). Unsuccessful strategies from S_{q0} and S_{q1} approach the goal position, but are too slow to complete the task in 180 seconds.

6.2 Using knowledge of the control task

Here we restrict qualitative functions to linear and quadratic functions and use the knowledge of the control task to find constraints among parameters of the quantitative strategy. In order to do this, we define quantitative strategies S_{q1} as a subset of strategies S with the properties:

1. *quadratic M_x^+ and M_x^- and linear $M_{\phi 1}^+$ and $M_{\phi 2}^+$.* Qualitative parameters, i.e. functions $M_{\phi 1}^+(\phi)$ and $M_{\phi 2}^+(\phi)$ are restricted to linear functions $k\phi + n$ and functions $M_x^+(X)$ and $M_x^-(X)$ are restricted to quadratic (or linear, when p=0) functions $px^2 + rx + m$. Parameters n and m are (without loss of generality) joined into DX_{mid}. This defines a subset of quantitative strategies of the form:

$$\begin{aligned} &\text{if } X < X_{mid} \text{ then} \\ &\quad DX_{des} = DX_{mid1} + p_1 x^2 + r_1 x + k_1 \phi \\ &\text{else } DX_{des} = DX_{mid2} + p_2 x^2 + r_2 x + k_2 \phi \end{aligned} \tag{8}$$

2. *Satisfy the task requirements* (obey task limits, stay at the goal position).

3. *Start towards goal* ($DX_{des}(0, 0) > 0$).

4. *Smooth and fast at* X_{mid} ($DX_{des}(X_{mid}-\epsilon,0) = DX_{des}(X_{mid}+\epsilon,0) = DX_{max}$).

Each of these properties imposes constraints on the parameters. The derivation given in Appendix yields in the strategies S_{q1} with the form given by rule 8 and constraints:

$$
\begin{aligned}
& 0 < X_{mid} < X_g \ , \ 0 < DX_{mid1} < DX_{max} \\
& -\frac{DX_{max}}{\phi_{max}} < k_1 = k_2 < \frac{DX_{max}}{\phi_{max}} \\
& -\frac{DX_{max} - DX_{mid1}}{X_{mid}^2} < p_1 < \frac{DX_{max} - DX_{mid1}}{X_{mid}^2} \\
& -\frac{DX_{max}}{(X_g - X_{mid})^2} < p_2 < \frac{DX_{max}}{(X_g - X_{mid})^2} \\
& r_1 = \frac{DX_{max} - DX_{mid1}}{X_{mid}} - p_1 X_{mid} \\
& r_2 = \frac{DX_{max}}{X_{mid} - X_g} - p_2(X_{mid} + X_g) \\
& DX_{mid2} = -X_g(X_g p_2 + r_2)
\end{aligned}
\tag{9}
$$

Further we define strategies S_{q2} as a subset of strategies S_{q1} with additional property that *the trolley travels fast*. This constraint simply requires that the speed of the trolley is large enough, when the trolley is far from the goal position. Constraints on parameters for strategies S_{q2} are the same as for S_{q1} with additional constraint on p_2, derived in Appendix.

Note that the last two constraints (*smooth and fast at* X_{mid} and *the trolley travels fast*) are not necessary. If we do not use them, strategies are still successful in terms of reaching the goal position under the task limits, but some of them are too slow and fail to conclude the task in the required time of 180 seconds.

To investigate how successful those strategies are, we tested strategies $S_{q1}(X_{mid}, DX_{mid1}, k_1, p_1, p_2)$ with different parameters: 2 for k_1 and DX_{mid1}, 3 for X_{mid} and 5 for p_1 and p_2. Parameters were taken uniformly spaced from the parameter's range and are given in Appendix. These parameter settings give 300 different qualitative strategies from S_{q1}. 180 of these strategies obey additional constraint *the trolley travels fast* and belong to S_{q2}. All of these 180 different quantitative strategies are successful and very fast. The best time is 38 sec., mean time 48 sec. and the worst time 78 sec. Considering the fact that the fastest operator's trace completes the task in 51 sec., the results are excellent. Out of the other 120 strategies, which do not respect the constraint *the trolley travels fast* and do not belong to S_{q2}, 100 are successful with mean time 84 sec. The details are given in Fig. 5.

7 Test of robustness

The strategies S_{q0}, were also tested for their robustness against changes in the start state, the test where the controllers in the form of decision or regression trees usually fail. In these experiments X_{mid} was set to values 15, 30, 45 and k_1 to $DX_{max}/4$. For each of these parameters settings, we generated 10 sets of random functions f_1^+, h_1^+, h_2^+, h_3^- with $R = 10$. In this way we get 30 quantitative strategies. All of these strategies were tested for their robustness against changes in the start state. Starting horizontal position was modified by 10% and 50%, and starting rope angle by -20%, 0% and 20%. This amounts to six combinations – six different tasks. Since controllers based on generalized operator's trajectory are goal directed, the change of the start state didn't affect their performance significantly. Out of $30 \times 6 = 180$ combinations, 162 were successful. The percentage of successful strategies with the changed start state is the same as the percentage of successful strategies with the original start state.

These results show significant improvement in the robustness of controllers using the generalized operator's trajectory over the controllers in the form of decision or regression trees.

8 Discussion and conclusion

Our approach involves learning separately (a) the system's dynamics and (b) the generalized trajectory the operator is trying to follow. The system's dynamics is learned using locally weighted regression. The trajectory is learned in the form of differential equations. Once the trajectory is known, a controller can easily be constructed since we know the desired next state and a local linear model near the current state. The induced control strategy often concludes the task faster than the original operator's traces. It is robust with respect to the changes in the system's dynamics, since the system's dynamics is learned apart from the trajectory. The induced strategy, in the form of the trajectory gives good insight in the human's subcognitive control strategy. Qualitative strategy, generalized from the operator's trajectory, is comprehensible and offers the possibility to improve the operator's control strategy.

Some specific points of interest are:

1. To understand the operator's control strategy, it is important to identify the subgoals that the operator is pursuing at various times. A method for identifying discrete subgoals was developed in [16]. The generalized operator's trajectory, applied in this paper, can be viewed as a continuously changing subgoal and enables skill reconstruction in more difficult domains. Moreover, it opens new perspectives to the reconstruction of human control skill, such as qualitative treatment.

2. In this paper we used methods of *symbolic* machine learning to generalize operators' trajectories. In principle, any approach to learning could be used to generalize trajectories, including subsymbolic learning such as artificial neural networks. However, using a symbolic approach has the advantage that the resulting generalized trajectory is expressed in explicit symbolic way and therefore easier to understand and analyze.

3. The induced strategy, in the form of the trajectory gives good insight in the operator's control strategy. Since it is expressed in a symbolic way, it is easy to understand, analyze and correct by considering the knowledge of the control task. One example of such analysis is given in this paper. We used the fact that the rope length at the goal position X_g should be L_g to correct the learned L_{des} trajectory.

4. Since the strategy as a generalized operator's strategy plus system's dynamics is goal directed, it is robust against change in the start state. Since the system's dynamics can be updated online, strategies are robust against change in the system's dynamics. Of course

this assumes that the operator's trajectory is still appropriate for the changed dynamics of the system.

5. The generalized operator's trajectory enables the derivation of qualitative control rules from the induced control strategy. We believe that qualitative rules are closer to human thinking about control strategies, are easier to understand and provide better insight into what the operator is trying to do.

6. Given a qualitative control strategy, we can tune a controller within the corresponding qualitative constraints. In this way a qualitative control strategy defines an optimisation space for better performance. As shown in this paper, we can also use some background knowledge and derive constraints on the tuned parameters. By tuning (giving the qualitative parameters like X_{mid}, *increasing function*, real numerical values) we can achieve better performance than that of the source quantitative strategy.

7. By using a learned qualitative strategy and considering knowledge of the control task to find constraints on qualitative parameters, the learned strategy can be used for considerably different tasks (different goal position, different safety requirements, different parameters of the simulator). For example we could use the learned qualitative strategy to move the load (possibly different) backwards from the goal position to the start position with more rigorous safety requirements (for example $DX < 1$m/s and $\phi < 6$ deg).

8. Regarding the derivation of a qualitative control strategy, one idea [1] is to induce qualitative rules directly from the execution trace. In this case a machine learning technique would be applied to a qualitative description of the execution trace. To use such an induced qualitative control strategy for controlling the system, the qualitative strategy would have to be operationalized by estimating numerical values of several parameters. The experimental results presented in this paper, and similar results in the Acrobot domain [17], show the potential of such an approach.

References

[1] I. Bratko. Qualitative reconstruction of control skill. In L. Ironi, editor, *Qualitative reasoning : 11th International Workshop*, pages 41–52. Pubblicazioni N. 1036. Univ. of Pavia: Instituto di analisi numerica, 1997. Cortona (Italy), June 3-6.

[2] I. Bratko, T. Urbančič, and C. Sammut. Behavioural cloning of control skill. In R.S. Michalski, I. Bratko, and M. Kubat, editors, *Machine Learning and Data Mining: Methods and Applications*, pages 335–351. John Wiley & Sons, Ltd, 1998.

[3] R. Camacho. Using machine learning to extract models of human control skill. In *Proceedings of AIT'95*, 1995. Brno, Czech Republic.

[4] R.A. Chambers and D. Michie. Man-machine co-operation in a learning task. *Computer Graphics: Techniques and Applications*, pages 179–186, 1969.

[5] W.S. Cleveland. Robust locally-weighted regression and smoothing scatterplots. *Journal of the American Statistical Association*, 74:829–836, 1979.

[6] P.E.K Donaldson. Error decorrelation studies on a human operator performing a balancing task. *Med. Electron. Biol. Engng.*, 2:393–410, 1964.

[7] S. Džeroski and L. Todorovski. Discovering dynamics. In *Proceedings of the 10th International Conference on Machine Learning*, pages 97–103. Morgan Kaufmann, 1993.

[8] K. Forbus. Qualitative process theory. *Artificial Intelligence*, 24:85–168, 1984.

[9] V. Križman, S. Džeroski, and B. Kompare. Discovering dynamics from measured data. *Electrotechnical Review*, 62:191–198, 1995.

[10] D. Michie. Knowledge, learning and machine intelligence. In L.S. Sterling, editor, *Intelligent Systems*, New York, 1993. Plenum Press.

[11] S. Muggleton. *Inductive Logic Programming - Logic Based Approach to ML*. Academic Press, San Diego, Ca, 1992. the A.P.I.C. series, no.38.

[12] C. Sammut, S. Hurst, D. Kedzier, and D. Michie. Learning to fly. In D. Sleeman and P. Edwards, editors, *Proceedings of the 9th International Workshop on Machine Learning*, pages 385–393. Morgan Kaufmann, 1992.

[13] S.A. Schaal and C.G. Atkeson. Assessing the quality of learned local models. In J.D. Cowan, G. Tesauro, and Alspector J., editors, *Advances in Neural Information Processing Systems*, volume 6. Morgan Kaufmann, 1994.

[14] T. Urbančič and I. Bratko. Reconstructing human skill with machine learning. In A. Cohn, editor, *Proceedings of the 11th European Conference on Artificial Intelligence*, pages 498–502. John Wiley & Sons, Ltd., 1994.

[15] D. Šuc. Reconstruction of skill based on goal structure, 1995. B.Sc.T., Faculty of Computer and Information Sciences, University of Ljubljana, Slovenia (in Slovenian).

[16] D. Šuc and I. Bratko. Skill reconstruction as induction of LQ controllers with subgoals. In Chris S. Mellish, editor, *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, volume 2, pages 914–920, 1997. Nagoya, Japan, August 1997.

[17] D. Šuc and I. Bratko. Skill modelling through symbolic and qualitative reconstruction of operator's trajectories, 1998. Tech.Report, Faculty of Computer and Information Sciences, Artificial Intelligence Laboratory, University of Ljubljana, Slovenia.

9 Appendix

9.1 Derivation of constraints on qualitative parameters

Here we derive constraints on qualitative parameters X_{mid}, M_x^+, M_x^-, $M_{\phi 1}^+$ and $M_{\phi 2}^+$ for quantitative strategies S_{q1} and S_{q2}, which have the general form given in 8. Each property of those strategies gives constraints on qualitative parameters:

1. *Quadratic M_x^+ and M_x^- and linear $M_{\phi 1}^+$ and $M_{\phi 2}^+$.* Since functions have to be increasing or decreasing we use constraints on their derivatives:

$$\underline{k_1, k_2 > 0}$$
$$2p_1 x + r_1 > 0 \text{ if } X < X_{mid}$$
$$\underline{r_1 > \max(0, -2p_1 X_{mid})}$$
$$2p_2 x + r_2 < 0 \text{ if } X > X_{mid}$$
$$\underline{r_2 < \min(-2p_2 X_g, -2p_2 X_{mid})}$$

2. *Satisfy the task requirements* (obey task limits, stay at the goal position). Because of the task limits ($\phi < |\phi_{max}|$,$DX < |DX_{max}|$) and since the trolley moves from its start position (X_0=0) to the goal position X_g:

$$-DX_{max} < k_1\phi, k_2\phi < DX_{max}$$
$$\underline{-\frac{DX_{max}}{\phi_{max}} < k_1, k_2 < \frac{DX_{max}}{\phi_{max}}}$$
$$\underline{0 < X_{mid} < X_g}$$

The final part of the task is to stay at the goal position for 8 seconds. To ensure this the following constraint must hold:

$$0 = DX_{des}(X_g, 0) = DX_{mid2} + p_2 X_g^2 + r_2 X_g + k_2 0 =$$
$$= DX_{mid2} + p_2 X_g^2 + r_2 X_g$$
$$\underline{DX_{mid2} = -X_g(p_2 X_g + r_2)}$$

3. *Start towards goal.* To ensure that the trolley starts to move toward the goal position $DX_{des}(0,0)$ must be positive. This gives the constraint:

$$DX_{des}(0,0) = DX_{mid1} + p_1 0 + r_1 + k_1 0$$
$$\underline{0 < DX_{mid1} < DX_{max}}$$

4. *Smooth and fast at X_{mid}.* Smooth control at X_{mid} is ensured by the constraint $\forall\phi : DX_{des}(X_{mid} - \epsilon, \phi) = DX_{des}(X_{mid} + \epsilon, \phi)$). So $\underline{k_1 = k_2}$. Under the condition of smooth control at X_{mid}, DX_{des} is clearly maximal at X_{mid} (after X_{mid} DX_{des} is decreasing). To achieve the best time, maximal DX_{des} (that is DX_{des} at X_{mid}) should be as large as possible, that is DX_{max} (ignoring ϕ).

$$DX_{des}(X_{mid} - \epsilon, 0) = DX_{des}(X = X_{mid} + \epsilon, 0) = DX_{max}$$
$$DX_{mid1} + p_1 {X_{mid}}^2 + r_1 X_{mid} = DX_{max} =$$
$$= DX_{mid2} + p_2 {X_{mid}}^2 + r_2 X_{mid}$$
$$\underline{r_1 = \frac{DX_{max} - DX_{mid1}}{X_{mid}} - p_1 X_{mid}}$$
$$DX_{max} = -{X_g}^2 p_2 - X_g r_2 + p_2 {X_{mid}}^2 + r_2 X_{mid}$$
$$DX_{max} = p_2(X_{mid}^2 - {X_g}^2) + r_2(X_{mid} - X_g)$$
$$\underline{r_2 = \frac{DX_{max}}{X_{mid} - X_g} - p_2(X_{mid} + X_g)}$$

Considering the constraints on functions derivatives this gives:

$$r_1 = \frac{DX_{max} - DX_{mid1}}{X_{mid}} - p_1 X_{mid} > \max(0, -2p_1 X_{mid})$$
$$r_2 = \frac{DX_{max}}{X_{mid} - X_g} - p_2(X_{mid} + X_g) < \min(-2p_2 X_g, -2p_2 X_{mid})$$

Let us consider all four possible cases:

- $\underline{p_1 \geq 0}$ then $\max(0, -2p_1 X_{mid}) = 0$

$$r_1 = \frac{DX_{max} - DX_{mid1}}{X_{mid}} - p_1 X_{mid} > 0$$
$$p_1 < \frac{DX_{max} - DX_{mid1}}{{X_{mid}}^2}$$

- $\underline{p_1 < 0}$ then $\max(0, -2p_1 X_{mid}) = -2p_1 X_{mid}$

$$r_1 = \frac{DX_{max} - DX_{mid1}}{X_{mid}} - p_1 X_{mid} > -2p_1 X_{mid}$$
$$p_1 > \frac{-(DX_{max} - DX_{mid1}}{{X_{mid}}^2}$$

- $\underline{p_2 \geq 0}$ then $\min(-2p_2 X_g, -2p_2 X_{mid}) = -120 p_2$

$$r_2 = \frac{DX_{max}}{X_{mid} - X_g} - p_2(X_{mid} + X_g) < -120 p_2$$
$$p_2 < \frac{DX_{max}}{(X_g - X_{mid})^2}$$

- $\underline{p_2 < 0}$ then $\min(-2p_2 X_g, -2p_2 X_{mid}) = -2p_2 X_{mid}$

$$r_2 = \frac{DX_{max}}{X_{mid} - X_g} - p_2(X_{mid} + X_g) < -2p_2 X_{mid}$$
$$p_2 > \frac{-DX_{max}}{(X_g - X_{mid})^2}$$

This gives the constraints on p_1 and p_2:

$$-\frac{DX_{max} - DX_{mid1}}{X_{mid}^2} < p_1 < \frac{DX_{max} - DX_{mid1}}{X_{mid}^2}$$
$$-\frac{DX_{max}}{(X_g - X_{mid})^2} < p_2 < \frac{DX_{max}}{(X_g - X_{mid})^2}$$

5. *The trolley travels fast.* To ensure that the trolley reaches the goal position fast, one additional constraint is needed. This constraint requires that the speed of the trolley is large enough ($DX_{des} > 0.02\, DX_{max}$) when the trolley is far from the goal position ($X \leq$

$0.98\,X_g$):

$$\begin{aligned}
&DX_{des}(X,0) > 0.02\,DX_{max} \text{ if } X \in [X_{mid}, 0.98\,X_g] \\
&DX_{des}(0.98X_g,0) > 0.02DX_{max} \text{ (since } DX_{des} \text{ is dec. in X)} \\
&DX_{mid2} + (0.98X_g)^2 p_2 + 0.98X_g r_2 > 0.02DX_{max} \\
&-X_g(p_2X_g + r_2) + (0.98X_g)^2 p_2 + 0.98X_g r_2 > 0.02DX_{max} \\
&(0.98^2-1)X_g p_2 50 - r_2 > \frac{DX_{max}}{X_g} \\
&-1.98X_g p_2 - \frac{DX_{max}}{X_{mid}-X_g} + p_2(X_{mid}+X_g) > \frac{DX_{max}}{X_g} \\
&p_2(-1.98X_g + X_{mid} + X_g) > \frac{DX_{max}}{X_g} + \frac{DX_{max}}{X_{mid}-X_g} \\
&\underline{p_2 < \frac{\frac{DX_{max}}{X_g} + \frac{DX_{max}}{X_{mid}-X_g}}{(X_{mid}-0.98X_g)}}
\end{aligned}$$

<u>To summarize the derivation of the constraints</u>:

- Quantitative strategies S_{q1} have the first four properties, giving the constraints on qualitative parameters:

$$\begin{aligned}
&0 < X_{mid} < X_g \ ,\ 0 < DX_{mid1} < DX_{max} \\
&-\frac{DX_{max}}{\phi_{max}} < k_1 = k_2 < \frac{DX_{max}}{\phi_{max}} \\
&-\frac{DX_{max} - DX_{mid1}}{{X_{mid}}^2} < p_1 < \frac{DX_{max} - DX_{mid1}}{{X_{mid}}^2} \\
&-\frac{DX_{max}}{(X_g - X_{mid})^2} < p_2 < \frac{DX_{max}}{(X_g - X_{mid})^2} \\
&r_1 = \frac{DX_{max} - DX_{mid1}}{X_{mid}} - p_1 X_{mid} \\
&r_2 = \frac{DX_{max}}{X_{mid} - X_g} - p_2(X_{mid} + X_g) \\
&DX_{mid2} = -X_g(X_g p_2 + r_2)
\end{aligned} \tag{10}$$

- Quantitative strategies $S_{q2} \subset S_{q1}$ also have the fifth property, giving additional constraint:

$$p_2 < \frac{\frac{DX_{max}}{X_g} + \frac{DX_{max}}{X_{mid}-X_g}}{(X_{mid}-0.98X_g)} \tag{11}$$

9.2 Parameters for testing strategies S_{q1}

To investigate how successful strategies S_{q1} (and S_{q2}) are, we tested them with different parameters: 2 for k_1 and DX_{mid1}, 3 for X_{mid} and 5 for p_1 and p_2. Parameters were taken uniformly spaced from the parameter's range given by eq. 10:

$$
\begin{aligned}
k_1 &= \frac{DX_{max}}{4\phi_{max}}, \frac{3DX_{max}}{4\phi_{max}} \\
DX_{mid1} &= \frac{DX_{max}}{3}, \frac{2DX_{max}}{3} \\
X_{mid} &= \frac{X_{max}}{4}, \frac{2X_{max}}{4}, \frac{3X_{max}}{4} \\
p_1 &= \frac{m_1}{5} \frac{DX_{max} - DX_{mid1}}{{X_{mid}}^2}, \; m_1 = -5, -3, 0, 3, 5 \\
p_2 &= \frac{m_2}{5} \frac{DX_{max}}{(X_g - X_{mid})^2}, \; m_2 = -5, -3, 0, 3, 5
\end{aligned}
\tag{12}
$$

Those parameter settings give $2 \times 2 \times 3 \times 5 \times 5 = 300$ different qualitative strategies from S_{q1}. 180 of those strategies obey additional constraint *the trolley travels fast* and belong to S_{q2}.