IJCNN 2019. International Joint Conference on Neural Networks. Budapest, Hungary. 14-19 July 2019

Optimal Pedestrian Evacuation in Building with Consecutive Differential Dynamic Programming

Yuanheng Zhu*, Haibo He†, Dongbin Zhao*, Zhongsheng Hou‡
*State Key Laboratory of Management and Control for Complex Systems
Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
Email: yuanheng.zhu@ia.ac.cn, dongbin.zhao@ia.ac.cn
†Department of Electrical, Computer, and Biomedical Engineering
University of Rhode Island, Kingston, RI 02881, USA
Email: he@ele.uri.edu
‡School of Automation, Qingdao University, Qingdao 266071, China
Email: zhshhou@bjtu.edu.cn

***Abstract*—Fast and efficient evacuation of pedestrians from an enclosed area is a difficult but crucial issue in modern society. In this paper, the optimization of evacuation from a building is studied. A graph is adopted to describe the building layout with nodes representing areas and edges representing connections. The dynamics of the evacuation process in the graph is formulated by a nonlinear discrete-time model at a macroscopic level. To find the optimal evacuation plan, a consecutive differential dynamic programming is developed. It inherits the differential dynamic programming property that solves the value and optimal policy locally. Additionally, it consecutively executes actions for multiple steps in the trajectory, which is beneficial to reduce computational burden and lower optimization difficulty. Simulations on a four-storey building layout demonstrates our method is efficient and suitable for on-site evacuation plan making.**

I. Introduction

With the increase of population and enrichment of recreation, people are now prone to crowd in certain places like movie theaters, stadiums, and subway stations, to attend social activities. Along with that is the fact that mass event has become a serious threat to human health and safety [1]. When people are crowded in an enclosed area, fire alarm, terrorist attack, or maybe just a sudden movement of someone will cause panic and create a rush for exits. Trampling and congestion occurs and may lead to serious injury or even casualty to evacuees. In recent years, frequency and severity of such events has increased gradually with years, and becomes a serious social issue for many places.

Through the study of praxeology and psychology, people in panic are easy to fall into the 'faster-is-slower' phenomena. When an enclosed area has few people, they feel free to move at their willing speed. However, with the increase of pedestrians, the inter-distance is shortened and people slow down their movement for a personal privacy consideration. When the density continues to increase, the movement will be seriously jammed and the flow comes to a complete stoppage. This phenomena is easily observed when two crowds run into a narrow area like doors or exits. In order to have a deep understanding, researchers study the pedestrian movement at a microscopic and a macroscopic levels separately. The famous microscopic model is the social force model that uses interactive forces between human and human, human and goal, and human and objectives, to govern the movement of individual agent [2]. For macroscopic models, researchers use density to describe the crowdedness, and apply fluid dynamic theory to the dynamics of pedestrian flow. In general, flow speed follows a decreasing function of density, and [3] gives a comprehensive study of the relationship.

After mounting pressure sensors at floor or installing cameras at ceiling, flow density and average velocity in an area can be fully detected with modern processing techniques. With the support of audio or visual display instructions, the behaviour of evacuees can be guided. Then it is possible to regulate the evacuation process to avoid jams and maximize the evacuation discharge. In [4], evacuation in a corridor is considered. The corridor area is partitioned into several parts and the system dynamics is described with a finite-dimensional ordinary differential equation. A calculus of variations method convert the problem to a two-point boundary value problem that is solved for the optimal control. [5] uses partial differential equation to describe the crowd model in one dimension. Three control models are proposed to avoid jams and shocks. Their feedback control is in a distributed setting in contrast to [4] that is discretized into different sections. To ensure the maximum discharge, [6] formulates a linear programming subject to control constraints so that the system tracks the critical density in all sections. They further extend the work to a network of corridors that uses nodes and edges to describe the connection and layout of evacuation route [7]. Still penetration rate and flow speed are determined by a linear programming that tries to make each node and edge track their critical states.

In the view of optimal control, the objective is to maximize the sum of costs of the system over a finite or infinite steps, and reinforcement learning (RL) has been proved to be a powerful tool to that [8], [9]. The optimal solution follows

This work was supported in part by the National Natural Science Foundation of China (NSFC) under grants No. 61603382, 61573353, 61533017, and was also supported in part by the National Science Foundation under grant CMMI 1526835.

978-1-7281-2009-6/$31.00 paper N-19916.pdf

the Bellman's optimality principle, so some RL methods try to solve the Bellman equation over the state space, which is quite computationally expensive. Differential dynamics programming (DDP) is a RL method that approximates the optimal solution locally. It uses second partial derivatives of value, cost, and dynamics around the nominal trajectory, and tries to find the improved control policy sequence to generate a better candidate solution. Through iteration, values are approximated more and more accurately, and the trajectory is improved closer and closer to the optimal one. To deal with control limits, [10] proposes a projected Newton method to search the improved control policy in the constrained control space. Compared to other approaches that tries to solve the Bellman equation, DDP approximates values locally, so it avoids the curse of dimensionality and is suitable to problems with high dimensions. In [11], authors use iLQG, a variant of DDP, as the guided policy to the end-to-end deep reinforcement learning (DRL) of deep visuomotor policy and the results show promising performance for manipulating complicated robots.

In this paper, we study the optimal evacuation of people in a building at a macroscopic level. The building layout is described by a graph. The areas are represented by nodes and the connections between areas are specified with edges. The state of the system is composed of pedestrian masses and densities in different nodes, and the control vector includes penetration rates and maximum allowed flow speeds. To lower optimization difficulty, a control action is executed for consecutive steps and a consecutive DDP algorithm is developed to optimize the control sequence. Through simulation experiments, it is demonstrated that the method provides an efficient way to make on-site optimal evacuation plan based on building and evacuation conditions.

II. Modeling Pedestrian Evacuation in Building

A. Graph representation of building layout

According to space partitions, a building layout can be generally subdivided into following types of areas: rooms, passages, staircases, and exits. A room has the capacity for a number of people and is connected to a passage via a door. Passages and staircases are areas that allow the movement of people. Exits are connections of building interior and the outside world. It is reasonable that every area is not isolated, but connected to other areas and finally to the exits. During the evacuation process, people will follow an escape route and move to the building outside.

To describe the building layout efficiently, a graph is adopted to represent the partitioned areas and connections between them. A graph is composed of nodes and edges, denoted by $\mathbb{G} = \{\mathbb{V}, \mathbb{E}\}$. Each node represents an area, and each edge specifies the connection of two nodes $n, n' \in \mathbb{V}$ that are adjacent in layout, denoted by (n, n'). Note that the edge does not include any distance information, but just the effective width of the connection area. According to the area characteristics, there are three kinds of nodes in $\mathbb{V}$: the source set $\mathbb{V}_S$, the corridor set $\mathbb{V}_C$, and the exit set $\mathbb{V}_E$. The source node $n_s \in \mathbb{V}_S$ corresponds to areas like classroom or laboratory where a certain number of people assemble for study or work. The corridor node $n_c \in \mathbb{V}_C$ constitutes an evacuation route for people to move. The exit node $n_e \in \mathbb{V}_E$ can be seen as a special corridor node that is the connection between the building interior and the outside world. Exit nodes naturally have output discharge to the outside.

Based on the category of nodes, there are three kinds of edges in $\mathbb{E}$: source-to-corridor (S2C) edge (n_s, n_c), corridor-to-corridor (C2C) edge (n_c, n_c), and corridor-to-exit (C2E) edge (n_c, n_e). In this work, it is assumed that S2C edge is unidirectional so that people only move from source to corridor, and C2C/C2E edge is bidirectional so that tail-to-head or head-to-tail directions are all valid. For two nodes n and n', if there exists an edge (n, n') in $\mathbb{E}$, it is said that n' is a neighbor of n, and vice versa. The neighbor set $\mathcal{N}(n)$ of n denotes the set of all nodes in $\mathbb{V}$ that are neighbors of n. The neighbor set of a source is composed of corridors. The neighbor set of a corridor may includes sources, corridors, and exits. The neighbor set of an exit is composed of corridors.

Fig. 1a presents the layout of the second floor of Morrill Hall in the University of Rhode Island (URI). There are four rooms that are used as classrooms and laboratories. They are marked in the figure and considered as the source nodes. Along the passage there are three staircases, separately located at both ends and the middle position. The graph representation of the second floor is given in Fig. 1b. We use squares to represent sources and circles to represent corridors and exits. The whole building has four floors. Each floor has basically the same layout but the middle staircase does not reach the forth floor. Ends of the three staircases are the exits. The graph of the whole building is given in Fig. 1c. All nodes are fully connected. When an evacuation signal is given, people in the building will follow a sequence of connected nodes to the outside.

B. Pedestrian flow model

After representing a building with a graph, the current state of evacuation process is composed of states in each node. In the next, we separately introduce the dynamics of source, corridor, and exit.

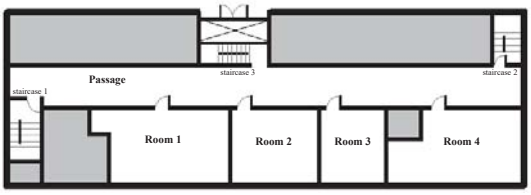
A source $n_s \in \mathbb{V}_S$ is a compartment of the building, so its core state is the pedestrian mass (number of people), denoted by P_s. The evolution of P_s is determined by the sum of its discharges to the neighbor corridors

$$\dot{P}_s = - \sum_{n' \in \mathcal{N}(n_s)} q_{(n_s, n')}. \tag{1}$$

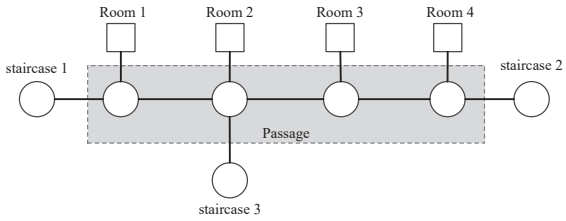
To distinguish with other discharges, $q_{(n_s,n')}$ is termed as penetration rate, and each edge corresponds to a value $\eta_{(n_s,n')}$

$$q_{(n_s,n')} = \eta_{(n_s,n')}. \tag{2}$$

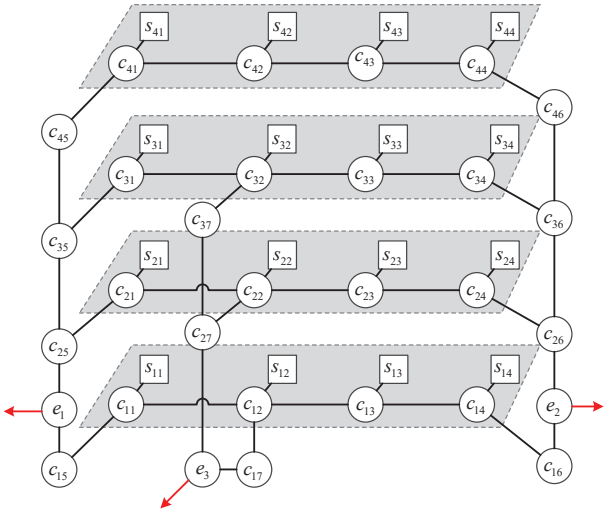
By controlling $\eta_{(n_s,n')}$, one can regulates the dynamics of pedestrian mass in sources. Unfortunately, due to the limit of effective width between two nodes, $\eta_{(n_s,n')}$ is upper bounded. In [3], a maximum penetration rate is suggested with 1.3 Person/s/m of effective width.



(a) Layout of the 2nd floor.



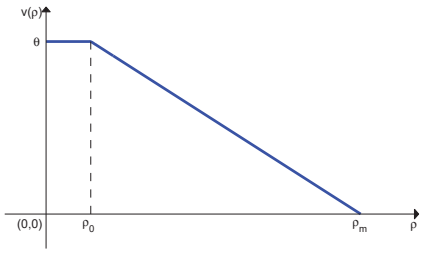
(b) Graph representation of the 2nd floor.



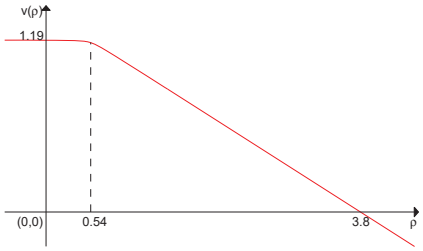
(c) Graph representation of the whole building.

Fig. 1: Layout and graph representation of Morrill Hall in URI.

The movement of people in an area can be considered as a flow, so fluid dynamic theory is applicable. The core state of corridors and exits is density ρ, which describes the degree of crowdedness of the area. The flow speed μ can be seen as a function of ρ. The relationship between μ and ρ has been studied thoroughly in literature. The consensus is that the flow speed is a decreasing function of density. [3] provides a function for flow speed, and has been used in the analysis of many evacuation cases. The function follows the curve given in Fig. 2a, where θ indicates the maximum possible flow speed. When density is lower than ρ_0, people can move freely at their willing speed. When ρ exceeds ρ_0, the flow speed decreases linearly with the increase of ρ, and finally comes to a complete jam at ρ_m. In [3], it is suggested that the maximum value of θ is bounded by $\theta_{\max} = 1.19$ m/s, and $\rho_0 = 0.54$ Persons/m^2, $\rho_m = 3.8$ Persons/m^2. To facilitate our study, we use a smooth



(a) Suggested relationship between evacuation speed and density in [3].



(b) Smooth approximated function to $\mu(\rho)$.

Fig. 2: Evacuation speed as a function of density.

function to approximate the curve with

$$\mu(\rho) = \theta h(\rho) = \theta \left[1 - \operatorname{smax}\left(\frac{\rho - \rho_0}{\rho_m - \rho_0}, p_1\right)\right] \tag{3}$$

where $\operatorname{smax}(z, p)$ is the smooth form of maximum function $\max(0, z)$ with

$$\operatorname{smax}(z, p) = \frac{1}{2}\left(\sqrt{z^2 + p^2} + z\right). \tag{4}$$

Under $\theta = 1.19$ and $p_1 = 0.1$, the smooth speed function with respect to density is plotted in Fig. 2b.

For a corridor node n_c, the core state is the density ρ_c and its dynamics is governed by the sum of discharges connected to n_c

$$\dot{\rho}_c = -\frac{1}{A_c} \sum_{n' \in \mathcal{N}(n_c)} q_{(n_c, n')} \tag{5}$$

where A_c is the effective area of n_c. If $n' \in \mathbb{V}_S$, the discharge is the penetration rate between n' and n_c

$$q_{(n_c, n')} = -\eta_{(n', n_c)}. \tag{6}$$

If $n' \in \{\mathbb{V}_C \cup \mathbb{V}_E\}$, the inter-node discharge is the product of density, flow speed, and effective width according to fluid dynamic theory. Since the movement in (n_c, n') is bidirectional, so we have

$$q_{(n_c, n')} = \begin{cases} \mu_{(n_c, n')} \rho_c W_{(n_c, n')} & \text{if } \mu_{(n_c, n')} \geq 0 \quad (7) \\ -\mu_{(n', n_c)} \rho' W_{(n', n_c)} & \text{if } \mu_{(n', n_c)} > 0 \quad (8) \end{cases}$$

where $\mu_{(n_c, n')}$ indicates the flow speed from n_c to n'. Then it holds that $\mu_{(n_c, n')} = -\mu_{(n', n_c)}$. $W_{(n_c, n')}$ and $W_{(n', n_c)}$ indicate the effective width of the connection between n_c and

n', and they have the same value. ρ' is the density in n'. In this paper, we assume that there can exist only one direction of flow between two nodes at one step. The discharge can be rewritten as

$$\begin{aligned} q_{(n_c,n')} =& \max(0, \mu_{(n_c,n')}\rho_c W_{(n_c,n')}) \\ &- \max(0, \mu_{(n',n_c)}\rho' W_{(n',n_c)}). \end{aligned} \tag{9}$$

Based on the speed function given in (3), a smooth form of discharge is obtained

$$\begin{aligned} q_{(n_c,n')} =& \operatorname{smax}(\theta_{(n_c,n')}h(\rho_c)\rho_c W_{(n_c,n')}, p_2) \\ &- \operatorname{smax}(\theta_{(n',n_c)}h(\rho')\rho' W_{(n',n_c)}, p_2) \end{aligned} \tag{10}$$

where $-\theta_{\max} \le \theta_{(n_c,n')} = -\theta_{(n',n_c)} \le \theta_{\max}$.

For an exit node $n_e \in \mathbb{V}_E$, in addition to the discharges $q_{(n_e,n')}$ to its neighbors, there is an output discharge q_e to the outside of the building, so

$$\dot{\rho}_e = -q_e - \sum_{n' \in \mathcal{N}(n_e)} q_{(n_e,n')} \tag{11}$$

where $q_{(n_e,n')}$ follows the formula given in (10). When people are in exits, it is nature for them to move at their maximum speed so that they can escape the building as soon as possible. Therefore, the outside discharge q_e has

$$q_e = \theta_{\max} h(\rho_e)\rho_e W_e \tag{12}$$

where W_e is the effective width of exit.

C. System dynamics and optimization objective

Now the whole system dynamics of pedestrian evacuation in a graph is described as follows. The system state is composed of all source pedestrian masses, corridor densities, and exit densities

$$x = [P_1, \dots, P_{|\mathbb{V}_S|}, \rho_1, \dots, \rho_{|\mathbb{V}_C|+|\mathbb{V}_E|}]^T \tag{13}$$

and the control vector includes penetration rates in S2C edges and maximum allowed speeds in C2C/C2E edges

$$u = [\dots, \eta_{(*,*)}, \dots, \theta_{(*,*)}, \dots]^T \tag{14}$$

where $(*,*)$ indicates arbitrary edges in $\mathbb{E}$. Note that for bidirectional C2C and C2E edges, only one direction of flow speed is stored in u, since the other direction speed has the same value but the opposite sign.

Combined with the above node model given in (1), (5), and (11), the system continuous-time dynamics can be described by a nonlinear function $\dot{x} = f(x,u)$, and the control variables are limited with

$$0 \le \eta_{(*,*)} \le \eta_{\max}, -\theta_{\max} \le \theta_{(*,*)} \le \theta_{\max}. \tag{15}$$

In order to simulate in computers, zero-order holder discretization is usually adopted to convert the continuous-time system to a discrete-time system with

$$x_{k+1} = F(x_k, u_k) \approx x_k + \delta t f(x_k, u_k) \tag{16}$$

where k is the discrete-time step index and δt is the sampling time. For the sake of simulation accuracy, δt generally selects small values (e.g. 0.1s herein).

For finite-time horizontal optimal control, the objective is to minimize the *total cost* of the system starting from an initial x_0

$$J(x_0) = \sum_{k=0}^{N-1} c(x_k, u_k) + c_f(x_N). \tag{17}$$

$c(x_k, u_k)$ specifies the cost at each step with respect to state and control, and $c_f(x_N)$ specifies the final cost with terminal state. In the evacuation process, the biggest concern is to evacuate pedestrians as fast as possible. At each step, the number of people still staying in the building is

$$\sum_{n_s \in \mathbb{V}_S} |P_s| + \sum_{n_c \in \{\mathbb{V}_C \cup \mathbb{V}_E\}} |\rho_c A_c|. \tag{18}$$

The absolute value operator is adopted to avoid the pedestrian masses and densities being negative. In order to smoothen the cost, a smooth absolute function is adopted with

$$\operatorname{sabs}(z,p) = \sqrt{z^2 + p^2} - p. \tag{19}$$

In this way, the cost function at each time step is defined

$$\begin{aligned} c(x,u) = &\sum_{n_s \in \mathbb{V}_S} \operatorname{sabs}(P_s, p_3) + \sum_{n_c \in \{\mathbb{V}_C \cup \mathbb{V}_E\}} \operatorname{sabs}(\rho_c A_c, p_3) \\ &+ w_u u^T u \end{aligned} \tag{20}$$

where w_u is the weight coefficient for control. The final cost is selected similarly

$$c_f(x) = \sum_{n_s \in \mathbb{V}_S} \operatorname{sabs}(P_s, p_3) + \sum_{n_c \in \{\mathbb{V}_C \cup \mathbb{V}_E\}} \operatorname{sabs}(\rho_c A_c, p_3). \tag{21}$$

Based on such definitions, the smaller $J(x_0)$ is minimized, the faster people escape from the building.

III. Differential Dynamic Programming

A. Differential dynamic programming

The aim of optimal control is to find a control sequence $\{u_0, \dots, u_N\}$ that minimizes the total cost $J(x_0)$. The *value* V_k is a function that specifies the minimum value for any state at step k. According to the Bellman's optimality principle, the value satisfies

$$V_k(x) = \min_u [c(x,u) + V_{k+1}(F(x,u))], 0 \le k \le N \tag{22}$$

$$V_{N+1}(x) = c_f(x). \tag{23}$$

Generally speaking, V_k is a complicated function and it is hard to give an analytic solution [12]. To address that, a quadratic form is adopted to approximate the value at a certain point.

Suppose we have had a sequence of control actions $\{u_0, \dots, u_N\}$, and following it we obtain a nominal state trajectory $\{x_0, \dots, x_{N+1}\}$. At step k, define the Q function

$$Q_k(x,u) = c(x,u) + V_{k+1}(F(x,u)). \tag{24}$$

Applying Taylor expansion to Q_k at nominal pair (x_k, u_k) we have[1]

$$Q_k(x,u) \approx \mathrm{Q}_k + \mathrm{Q}_{x,k}^T \delta x_k + \mathrm{Q}_{u,k}^T \delta u_k + \delta u_k^T \mathrm{Q}_{ux,k} \delta x_k + \frac{1}{2}\delta x_k^T \mathrm{Q}_{xx,k} \delta x_k + \frac{1}{2}\delta u_k^T \mathrm{Q}_{uu,k} \delta u_k \quad (25)$$

with $\delta x_k = x - x_k$, $\delta u_k = u - u_k$, and

$$\mathrm{Q}_k = \mathrm{c}_k + \mathrm{V}_{k+1} \quad (26)$$
$$\mathrm{Q}_{x,k} = \mathrm{c}_{x,k} + \mathrm{F}_{x,k}^T \mathrm{V}_{x,k+1} \quad (27)$$
$$\mathrm{Q}_{u,k} = \mathrm{c}_{u,k} + \mathrm{F}_{u,k}^T \mathrm{Q}_{x,k+1} \quad (28)$$
$$\mathrm{Q}_{xx,k} = \mathrm{c}_{xx,k} + \mathrm{F}_{x,k}^T \mathrm{V}_{xx,k+1} \mathrm{F}_{x,k} + \mathrm{V}_{x,k+1} \cdot \mathrm{F}_{xx,k} \quad (29)$$
$$\mathrm{Q}_{ux,k} = \mathrm{c}_{ux,k} + \mathrm{F}_{u,k}^T \mathrm{V}_{xx,k+1} \mathrm{F}_{x,k} + \mathrm{V}_{x,k+1} \cdot \mathrm{F}_{ux,k} \quad (30)$$
$$\mathrm{Q}_{uu,k} = \mathrm{c}_{uu,k} + \mathrm{F}_{u,k}^T \mathrm{V}_{xx,k+1} \mathrm{F}_{u,k} + \mathrm{V}_{x,k+1} \cdot \mathrm{F}_{uu,k} \quad (31)$$

where the last items in (29)–(31) denote the contraction with a tensor.

To achieve the minimization at the right-hand side of (22), the optimal δu_k^* should be

$$\delta u_k^* = \arg\min_{\delta u_k} Q_k(x, u_k + \delta u_k) = \mathrm{k}_k + \mathrm{K}_k \delta x_k \quad (32)$$

with

$$\mathrm{k}_k = -\mathrm{Q}_{uu,k}^{-1} \mathrm{Q}_{u,k}, \mathrm{K}_k = -\mathrm{Q}_{uu,k}^{-1} \mathrm{Q}_{ux,k}. \quad (33)$$

Inserting the control policy into (22), the value at step k has quadratic approximation with

$$\mathrm{V}_k = \mathrm{Q}_k - \frac{1}{2}\mathrm{Q}_{u,k}^T \mathrm{Q}_{uu,k}^{-1} \mathrm{Q}_{u,k} \quad (34)$$
$$\mathrm{V}_{x,k} = \mathrm{Q}_{x,k} - \mathrm{Q}_{ux,k}^T \mathrm{Q}_{uu,k}^{-1} \mathrm{Q}_{u,k} \quad (35)$$
$$\mathrm{V}_{xx,k} = \mathrm{Q}_{xx,k} - \mathrm{Q}_{ux,k}^T \mathrm{Q}_{uu,k}^{-1} \mathrm{Q}_{ux,k}. \quad (36)$$

After obtaining V_k, we are able to proceed the backward pass to optimize the control policy for the $(k-1)$-th step. The whole process starts from $V_{N+1}(x_N) = c_f(x_N)$.

Note that in (32), the minimization is reached only if $\mathrm{Q}_{uu,k}$ is positive definite. But in many cases, the positivity is not guaranteed along the nominal trajectory. Therefore, regularization is necessary for the backward pass. In literature, two kinds of regularization is mostly used, and they can be combined together to redefine matrices with

$$\tilde{\mathrm{Q}}_{uu,k} = \mathrm{c}_{uu,k} + \mathrm{F}_{u,k}^T (\mathrm{V}_{xx,k+1} + \mu_1 \mathbf{I}) \mathrm{F}_{u,k} + \mathrm{V}_{x,k+1} \cdot \mathrm{F}_{uu,k} + \mu_2 \mathbf{I}$$
$$\tilde{\mathrm{Q}}_{ux,k} = \mathrm{c}_{ux,k} + \mathrm{F}_{u,k}^T (\mathrm{V}_{xx,k+1} + \mu_1 \mathbf{I}) \mathrm{F}_{x,k} + \mathrm{V}_{x,k+1} \cdot \mathrm{F}_{ux,k}.$$

The values of regularization parameters μ_1 and μ_2 can be adjusted dynamically to accelerate the learning process. Increase the parameters if the algorithm is divergent or the total cost is not reduced, and decrease otherwise. More detailed description on the adjustment of μ_1 and μ_2 is available in [13]. Then the calculation of control policy k_k and K_k are made with the regularized matrices

$$\mathrm{k}_k = -\tilde{\mathrm{Q}}_{uu,k}^{-1} \mathrm{Q}_{u,k}, \mathrm{K}_k = -\tilde{\mathrm{Q}}_{uu,k}^{-1} \tilde{\mathrm{Q}}_{ux,k}. \quad (37)$$

Based on that, the update of values is improved to cancel matrix inversion calculations

$$\mathrm{V}_k = \mathrm{Q}_k + \frac{1}{2}\mathrm{k}_k^T \mathrm{Q}_{uu,k} \mathrm{k}_k + \mathrm{k}_k^T \mathrm{Q}_{u,k} \quad (38)$$
$$\mathrm{V}_{x,k} = \mathrm{Q}_{x,k} + \mathrm{K}_k^T \mathrm{Q}_{uu,k} \mathrm{k}_k + \mathrm{K}_k^T \mathrm{Q}_{u,k} + \mathrm{Q}_{ux,k}^T \mathrm{k}_k \quad (39)$$
$$\mathrm{V}_{xx,k} = \mathrm{Q}_{xx,k} + \mathrm{K}_k^T \mathrm{Q}_{uu,k} \mathrm{K}_k + \mathrm{K}_k^T \mathrm{Q}_{ux,k} + \mathrm{Q}_{ux,k}^T \mathrm{K}_k. \quad (40)$$

After the backward pass, we have a new sequence of control policies $\{\mathrm{k}_k, \mathrm{K}_k\}$. When executing them through a forward pass, it has a high probability to reduce the total cost. Because polices are generated around the last nominal pair (x_k, u_k), if the system deviates too far from the original trajectory, the improvement effect may disappear. To overcome that, a backtracking method is adopted to linear search the best solution with $0 < \alpha \leq 1$ following

$$\hat{u}_k = u_k + \alpha \mathrm{k}_k + \mathrm{K}_k (\hat{x}_k - x_k) \quad (41)$$
$$\hat{x}_{k+1} = F(\hat{x}_k, \hat{u}_k). \quad (42)$$

Once the best candidate solution is obtained, take it as the new nominal trajectory and repeat the backward pass and forward pass to further optimize the control sequence. The algorithm stops when the reduction of total cost is lower than a small threshold.

B. Control limitation

In the above description of DDP algorithm, the control limit is not considered in both backward pass and forward pass. In fact, in the evacuation process, pedestrian rate and flow speed are bounded, generalized into the element-wise constraint $\underline{u} \leq u \leq \overline{u}$. The minimization in the right-hand side of (22) becomes a constrained optimization. Based on the quadratic approximation and letting $q = \mathrm{Q}_{u,k} + \mathrm{Q}_{ux,k} \delta x_k$, $H = \mathrm{Q}_{uu,k}$, $z = \delta u_k$, $\underline{z} = \underline{u} - u_k$, and $\overline{z} = \overline{u} - u_k$, the problem is formulated as a constrained quadratic programming

$$\begin{aligned} z^* = \arg\min \quad & q^T z + \tfrac{1}{2} z^T H z \\ s.t. \quad & \underline{z} \leq z \leq \overline{z}. \end{aligned} \quad (43)$$

Using the box constraint characteristics, the problem can be effectively solved by projected Newton search method [10], [14].

Given an initial guess of z (e.g. $z = 0$), calculate the gradient of objective $g(z) = q + Hz$ and check the clamped and free dimensions in z

$$\mathsf{c}(z) = \left\{ j \in 1, \ldots, n \;\middle|\; \begin{array}{ll} z_j = \underline{z}_j, & g_j > 0 \\ z_j = \overline{z}_j, & g_j < 0 \end{array} \right\} \quad (44)$$
$$\mathsf{f}(z) = \{ j \in 1, \ldots, n \,|\, j \notin \mathsf{c}(z) \}. \quad (45)$$

For ease of analysis, rearrange the elements of vectors and matrix in the form

$$z = \begin{bmatrix} z_{\mathsf{f}} \\ z_{\mathsf{c}} \end{bmatrix}, q = \begin{bmatrix} q_{\mathsf{f}} \\ q_{\mathsf{c}} \end{bmatrix}, H = \begin{bmatrix} H_{\mathsf{ff}} & H_{\mathsf{fc}} \\ H_{\mathsf{cf}} & H_{\mathsf{cc}} \end{bmatrix}. \quad (46)$$

[1] We use italic symbols (e.g. Q_k) to indicate the function, and the non-italic symbols (e.g. Q_k) to indicate the function value at the nominal trajectory (e.g. $\mathrm{Q}_k = Q_k(x_k, u_k)$)

Therefore the free gradient in g is $g_{\mathsf{f}} = q_{\mathsf{f}} + H_{\mathsf{ff}} z_{\mathsf{f}} + H_{\mathsf{fc}} z_{\mathsf{c}}$, and the projected Newton step for whole z has

$$\delta z = \begin{bmatrix} -H_{\mathsf{ff}}^{-1} g_{\mathsf{f}} \\ 0 \end{bmatrix} = \begin{bmatrix} -z_{\mathsf{f}} - H_{\mathsf{ff}}^{-1}(q_{\mathsf{f}} + H_{\mathsf{fc}} z_{\mathsf{c}}) \\ 0 \end{bmatrix}. \quad (47)$$

We can use a line-search method to find a better candidate solution

$$\hat{z}(\alpha) = \min(\overline{z}, \max(\underline{z}, z + \alpha \delta z)). \quad (48)$$

The process is repeated for the new $\hat{z}$ until the reduction of objective is less than a small threshold.

The optimal solution to the constrained quadratic programming should equal boundary values at the clamped dimensions and zero gradient at the free dimensions, so we have

$$z_{\mathsf{f}}^* = -H_{\mathsf{ff}}^{-1} q_{\mathsf{f}} - H_{\mathsf{ff}}^{-1} H_{\mathsf{fc}} z_{\mathsf{c}}^*. \quad (49)$$

Recover the original notations with $z = \delta u_k$, $q = \mathrm{Q}_{u,k} + \mathrm{Q}_{ux,k}\delta x_k$, and $H = \mathrm{Q}_{uu,k}$, the optimized policy has

$$\mathrm{k}_k = \begin{bmatrix} -(\mathrm{Q}_{uu,k})_{\mathsf{ff}}^{-1}(\mathrm{Q}_{u,k})_{\mathsf{f}} - (\mathrm{Q}_{uu,k})_{\mathsf{ff}}^{-1}(\mathrm{Q}_{uu,k})_{\mathsf{fc}} z_{\mathsf{c}}^* \\ z_{\mathsf{c}}^* \end{bmatrix} \quad (50)$$

$$\mathrm{K}_k = \begin{bmatrix} -(\mathrm{Q}_{uu,k})_{\mathsf{ff}}^{-1}(\mathrm{Q}_{ux,k})_{\mathsf{ff}} & -(\mathrm{Q}_{uu,k})_{\mathsf{ff}}^{-1}(\mathrm{Q}_{ux,k})_{\mathsf{fc}} \\ 0 & 0 \end{bmatrix}. \quad (51)$$

In the forward pass, the new trajectory is generated with the constrained control sequence

$$\hat{u}_k = \min(\overline{u}, \max(\underline{u}, u_k + \alpha \mathrm{k}_k + \mathrm{K}_k(\hat{x}_k - x_k))). \quad (52)$$

IV. Consecutive DDP

The original DDP optimizes control actions at every step. In many cases, this may lead to huge computational burden. For instance, the evacuation process chooses step size 0.1s. If the system simulates 250s and adjusts control at every step, the number of actions that need to be optimized is 2500. If we adjust control actions every 50 steps, i.e. every 5 seconds, the control sequence is reduced to the length of 50. The system still maintains a satisfying performance, but the optimization burden and learning difficulty is greatly reduced. Motivated by that, consecutive control execution is adopted and a new DDP algorithm is developed.

Suppose at i-th step, a new control action u_i is calculated based on current state x_i. u_i is repeatedly executed for M consecutive steps. Then at $(i+M)$-th step, the system calculates a new u_{i+M} and execute it for the next M steps. Following (22), the Bellman equation now becomes

$$V_i(x_i) = \min_{u_i} \left[c(x_i, u_i) + \cdots + c(x_{i+N-1}, u_i) + V_{i+N}(x_{i+N}) \right] \quad (53)$$

with

$$x_{j+1} = F(x_j, u_i), i \le j < i + N. \quad (54)$$

Given the value V_{i+N}, let $Q_{i+N}(x,u) = V_{i+N}(x)$ and define quadratic Q functions for every $i \le j < i+N$ in a backward pass

$$\begin{aligned} Q_j(x,u) =& \mathrm{Q}_j + \mathrm{Q}_{x,j}^T \delta x_j + \mathrm{Q}_{u,j}^T \delta u_i + \delta u_i^T \mathrm{Q}_{ux,j} \delta x_j \\ &+ \frac{1}{2} \delta x_j^T \mathrm{Q}_{xx,j} \delta x_j + \frac{1}{2} \delta u_i^T \mathrm{Q}_{uu,j} \delta u_i \end{aligned} \quad (55)$$

with $\delta u_i = u - u_i$, $\delta x_j = x - x_j$, and

$$\begin{aligned} \mathrm{Q}_j =& \mathrm{c}_j + \mathrm{Q}_{j+1} \\ \mathrm{Q}_{x,j} =& \mathrm{c}_{x,j} + \mathrm{F}_{x,j}^T \mathrm{Q}_{x,j+1} \\ \mathrm{Q}_{u,j} =& \mathrm{c}_{u,j} + \mathrm{F}_{u,j}^T \mathrm{Q}_{x,j+1} + \mathrm{Q}_{u,j+1} \\ \mathrm{Q}_{xx,j} =& \mathrm{c}_{xx,j} + \mathrm{F}_{x,j}^T \mathrm{Q}_{xx,j+1} \mathrm{F}_{x,j} + \mathrm{Q}_{x,j+1} \cdot \mathrm{F}_{xx,j} \\ \mathrm{Q}_{ux,j} =& \mathrm{c}_{ux,j} + \mathrm{F}_{u,j}^T \mathrm{Q}_{xx,j+1} \mathrm{F}_{x,j} + \mathrm{Q}_{ux,j+1} \mathrm{F}_{x,j} \\ &+ \mathrm{Q}_{x,j+1} \cdot \mathrm{F}_{ux,j} \\ \mathrm{Q}_{uu,j} =& \mathrm{c}_{uu,j} + \mathrm{F}_{u,j}^T \mathrm{Q}_{xx,j+1} \mathrm{F}_{u,j} + \mathrm{Q}_{x,j+1} \cdot \mathrm{F}_{uu,j} \\ &+ \mathrm{Q}_{uu,j+1} + 2\mathrm{Q}_{ux,j+1} \mathrm{F}_{u,j}. \end{aligned} \quad (56)$$

After obtaining Q_i at the beginning of the M consecutive steps, generate the improved control policy following

$$\delta u_i^* = \arg\min_{\delta u_i} Q_i(x, u_i + \delta u_i) = \mathrm{k}_i + \mathrm{K}_i \delta x_i \quad (57)$$

where k_i and K_i are determined by (33) if no control limits, or (50), (50) with box constraints.

Still to ensure the positivity of $\mathrm{Q}_{xx,i}$ in the minimization, two regularization terms are introduced in the backward pass

$$\tilde{\mathrm{Q}}_{xx,i+N} = \mathrm{V}_{xx,i+N} + \mu_1 \mathbf{I} \quad (58)$$

$$\tilde{\mathrm{Q}}_{uu,i} = \mathrm{Q}_{uu,i} + \mu_2 \mathbf{I}. \quad (59)$$

k_i and K_i are generated with the regularized $\tilde{\mathrm{Q}}_{uu,i}$ and $\tilde{\mathrm{Q}}_{ux,i}$.

The value for the i-th step is approximated with (38)–(40). Then the backward pass is repeated with $V_i(x)$ for the previous $(i-M), \ldots, i$ steps. For ease of implementation, the total length of nominal trajectory is $(TM+1)$ and the backward pass starts with $V_{TM}(x) = c_f(x)$. In other words, there are T control actions that are to be optimized.

After obtaining the control policy sequence $\{\mathrm{k}_i, \mathrm{K}_i\}$ for steps $i = 0, M, \ldots, (T-1)M$, the forward pass tries to find a better nominal trajectory with

$$\begin{aligned} &\hat{u}_i = u_i + \alpha \mathrm{k}_i + \mathrm{K}_i(\hat{x}_i - x_i), i = 0, M, \ldots, (T-1)M \\ &\hat{x}_{j+1} = F(\hat{x}_j, \hat{u}_i), i \le j < i + M. \end{aligned} \quad (60)$$

V. Simulation Study

Now we consider the graph illustrated in Fig. 1c, and use the proposed consecutive DDP algorithm to learn the optimal evacuation plan. For simplicity, all corridor nodes have the area of 12.5 m^2. The exit nodes have the area of 12 m^2. The effective widths in C2C and C2E edges are 2.5 m, and the effective widths of exits to the outside are 1.5 m. The penetration rate has limit $0 \le \eta \le 1.3$ Person/s and the maximum flow speed is set to $\theta_{\max} = 1.19$ m/s. The smooth parameters select $p_1 = p_2 = p_3 = 0.1$.

In the simulation, each source node has an initial pedestrian mass 30 Persons, and each corridor node has a density of 0.1 Person/m^2. The consecutive DDP algorithm calculates the optimal evacuation plan. Once the evacuation is triggered, people will follow the audio/display instructions that give the optimal penetration rates and flow speeds, and move towards the building exits.

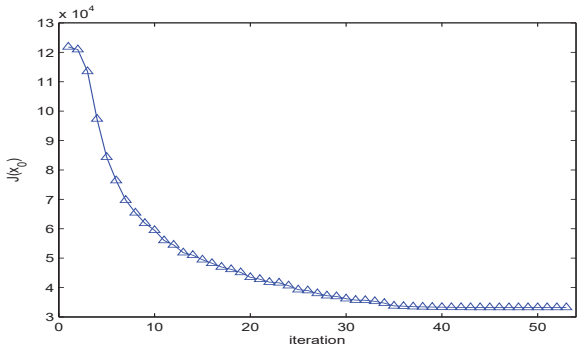


Fig. 3: Learning curve of consecutive DDP for case 1.

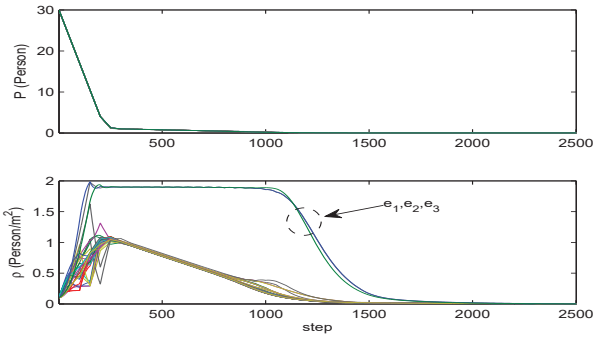


Fig. 5: Optimal state trajectory by consecutive DDP for case 1.

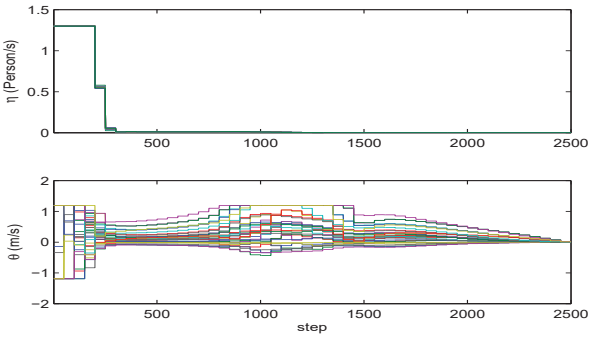


Fig. 4: Optimal control sequence by consecutive DDP for case 1.

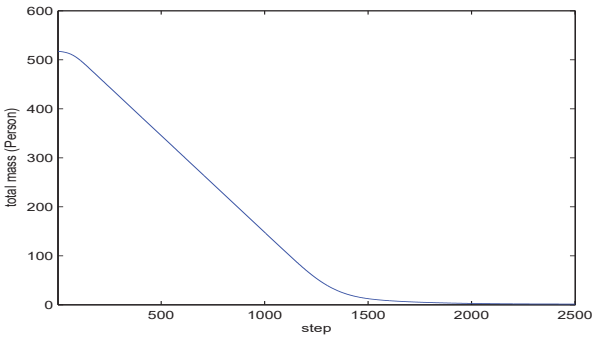


Fig. 6: Curve of pedestrian mass for case 1.

A. Case 1: All exits available

In the first experiment, three exits are all available for evacuees. The learning curve of consecutive DDP is given in Fig. 3. The total cost of the nominal trajectory at each iteration is shown in the plot. Through backward pass and forward pass, the total cost is reduced from the initial 1.22×10^5 to 3.32×10^4. The optimal control sequence is presented in Fig. 4. It is observed that actions are adjusted every 50 steps, which lowers the control frequency without deteriorating performance. It helps to decrease the optimization difficulty. All actions are within their limits. Following the evacuation plan, the system trajectory is plotted in Fig. 5. It is obvious that the pedestrian mass in the building is successfully evacuated. The optimal actions avoid the blockage that may happen. The optimal evacuation plan ensures the density of three exit nodes is maintained at 1.9 Person/m^2 so that the exit discharge is maximized with 1.98 Person/s. This requires the cooperation of interior nodes so that the input discharge and output discharge of exits are balanced. The reduction of total number of people is illustrated in Fig. 6. After a short initial phase, the reduction trends to be stable until the mass is close to zero.

B. Case 2: Two exits available

Next we consider the case that exit e_1 is blocked due to certain reasons like damaged or out of operation. Only exits e_2 and e_3 are available during the evacuation process, and e_1 becomes an ordinary corridor node that has no output discharge to the outside.

Based on the modified graph, our consecutive DDP learns a new optimal evacuation plan. In Figs. 7 and 8, the optimal control sequence and the corresponding state trajectory are presented. Due to the change of layout, the optimal actions for many nodes are changed. The curve of the total pedestrian mass in the building is shown in Fig. 9. Compared to Fig. 6, the evacuation time is obviously lengthened, but it still successfully evacuates the building without the occurrence of jams. It demonstrates that our algorithm is suitable to make on-site optimal evacuation plan according to building and pedestrian conditions.

VI. Conclusion

The optimization of building evacuation is studied in this paper. The introduction of graph representation makes it possible to describe building layout mathematically. The system of the evacuation process is established with states of all nodes, including pedestrian masses and densities. The control vector is composed of penetration rates and flow velocities in edges.

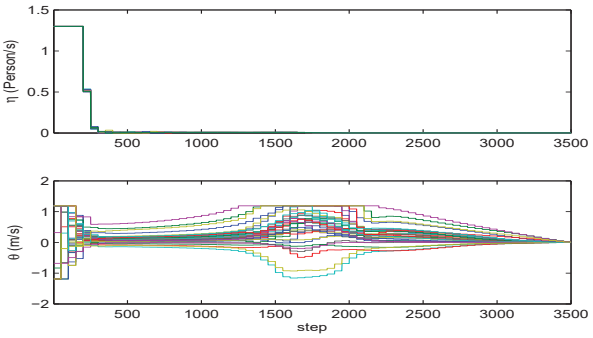


Fig. 7: Optimal control sequence by consecutive DDP for case 2.

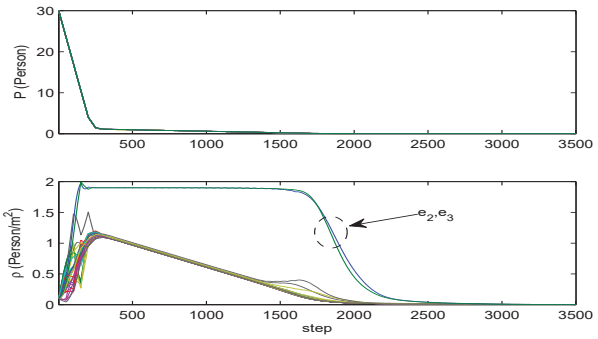


Fig. 8: Optimal state trajectory by consecutive DDP for case 2.

The optimization objective is to accelerate the evacuation, which is reflected in the minimization of total cost that is the sum of evacuees staying in the building over steps. To reduce optimization difficulty, control actions are executed consecutively for multiple steps, and consecutive DDP is developed to solve the optimal control problem with control limits. It is an online process since only the current state is inserted to the algorithm. In macroscopic simulations, the method has promising performance. Unfortunately, in real-world situations, the movement of pedestrians is individual and the evacuation process should be performed at the microscopic level. The future work is to shorten the gap between the macroscopic evacuation optimization and the microscopic real-world performance.

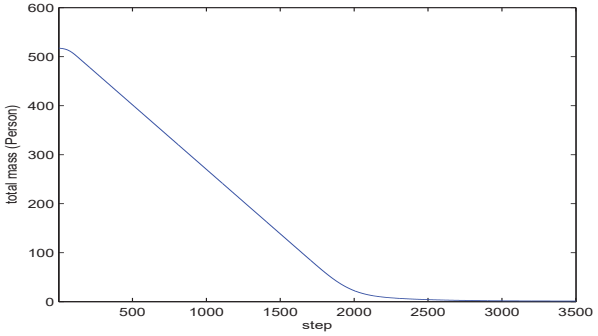


Fig. 9: Curve of pedestrian mass for case 2.

REFERENCES

[1] D. Helbing, L. Buzna, A. Johansson, and T. Werner, "Self-organized pedestrian crowd dynamics: Experiments, simulations, and design solutions," *Transportation Science*, vol. 39, no. 1, pp. 1–24, Feb. 2005.

[2] D. Helbing, I. Farkas, and T. Vicsek, "Simulating dynamical features of escape panic," *Nature*, vol. 407, no. 6803, p. 487, 2000.

[3] M. J. Hurley, D. T. Gottuk, J. R. Hall Jr, K. Harada, E. D. Kuligowski, M. Puchovsky, J. M. Watts Jr, and C. J. Wieczorek, *SFPE Handbook of Fire Protection Engineering*. Springer, 2015.

[4] A. Shende, P. Kachroo, C. K. Reddy, and M. Singh, "Optimal control of pedestrian evacuation in a corridor," in *2007 IEEE Intelligent Transportation Systems Conference*, Sept 2007, pp. 385–390.

[5] S. A. Wadoo and P. Kachroo, "Feedback control of crowd evacuation in one dimension," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 1, pp. 182–193, March 2010.

[6] A. Shende, M. P. Singh, and P. Kachroo, "Optimization-based feedback control for pedestrian evacuation from an exit corridor," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1167–1176, Dec 2011.

[7] ——, "Optimal feedback flow rates for pedestrian evacuation in a network of corridors," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1053–1066, Sept 2013.

[8] Y. Zhu, D. Zhao, X. Yang, and Q. Zhang, "Policy iteration for H_∞ optimal control of polynomial nonlinear systems via sum of squares programming," *IEEE Transactions on Cybernetics*, vol. 48, no. 2, pp. 500–509, Feb 2018.

[9] Y. Zhu, D. Zhao, and Z. Zhong, "Adaptive optimal control of heterogeneous CACC system with uncertain dynamics," *IEEE Transactions on Control Systems Technology*, 2018, DOI: 10.1109/TCST.2018.2811376.

[10] Y. Tassa, N. Mansard, and E. Todorov, "Control-limited differential dynamic programming," in *IEEE International Conference on Robotics and Automation*, 2014, pp. 1168 – 1175.

[11] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2015.

[12] Y. Zhu and D. Zhao, "Comprehensive comparison of online ADP algorithms for continuous-time optimal control," *Artificial Intelligence Review*, vol. 49, no. 4, pp. 531–547, 2018.

[13] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 4906–4913.

[14] Y. Zhu, D. Zhao, X. Li, and D. Wang, "Control-limited adaptive dynamic programming for multi-battery energy storage systems," *IEEE Transactions on Smart Grid*, 2018, DOI: 10.1109/TSG.2018.2854300.