Manipulation of Deformable Objects

Challenges, Strategies, Applications

Gianluca Palli

gianluca.palli@unibo.it

Department of Electrical, Electronic and Information Engineering
Laboratory of Automation and Robotics
University of Bologna
Viale del Risorgimento 2
40136 Bologna

S.I.D.R.A. PhD Summer School - Bertinoro 2021

Project Information

Robotic Technologies for the Manipulation of Complex Deformable Linear Objects

REMODEL

The research leading to the results presented in the following has received funding from the European Unions Horizon 2020 research and innovation program forthe project REMODEL - Robotic tEchnologies for the Manipulation of cOmplex DeformablE Linear objects - under grant agreement n 870133.

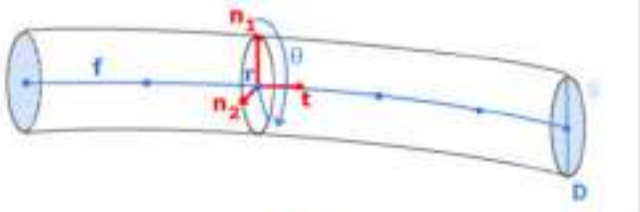
For more information, you can visit the project website: https://remodel-project.eu/

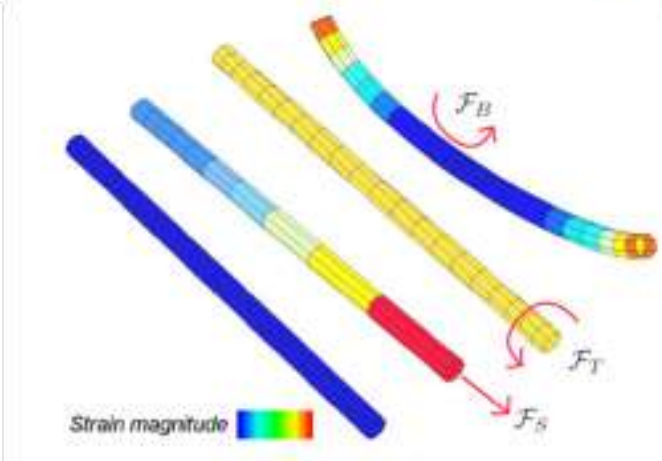
VIDEO

Table of contents

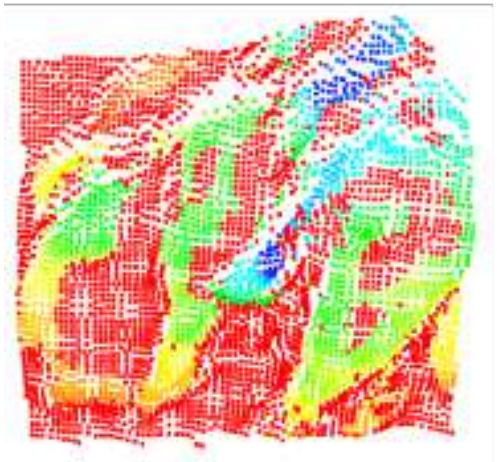
Deformable Objects Typologies

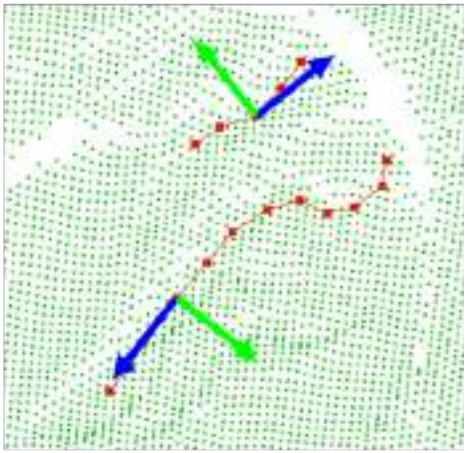
Deformable Linear Objects (DLOs)



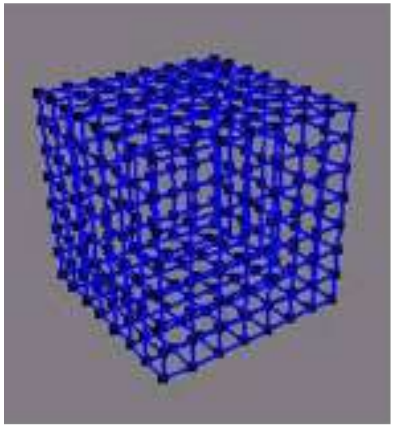


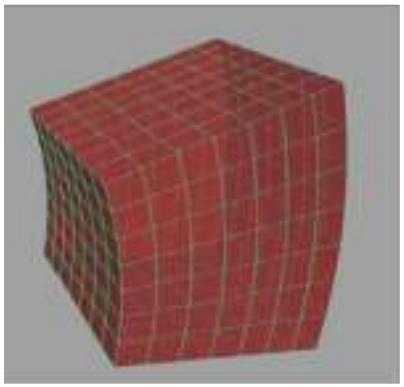
Deformable Planar Objects

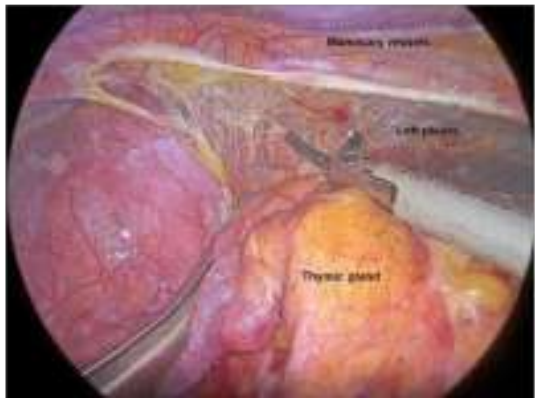




Deformable 3D Objects





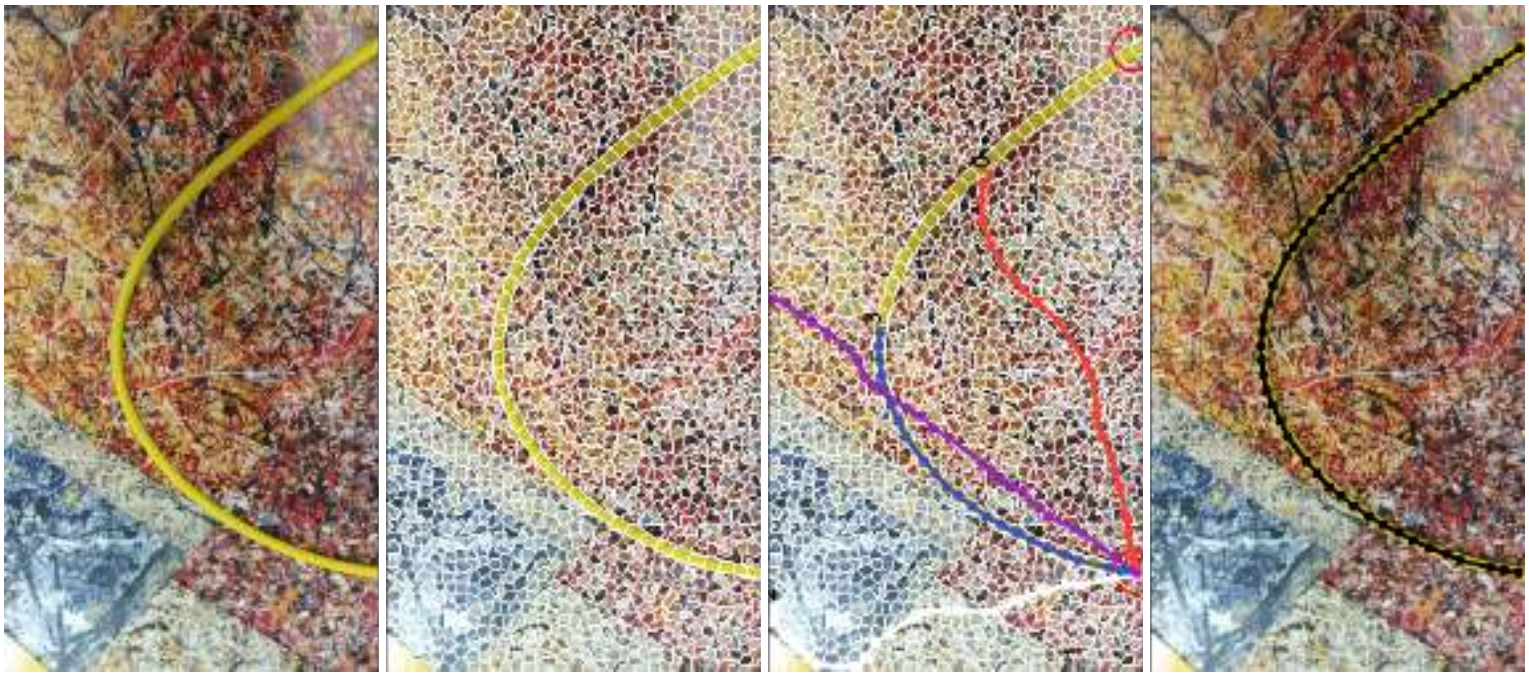


DLO State Perception in 2D

DLO Shape Detection

Ariadne Algorithm [De Gregorio et al., ACCV 2018]; Limitations due to:

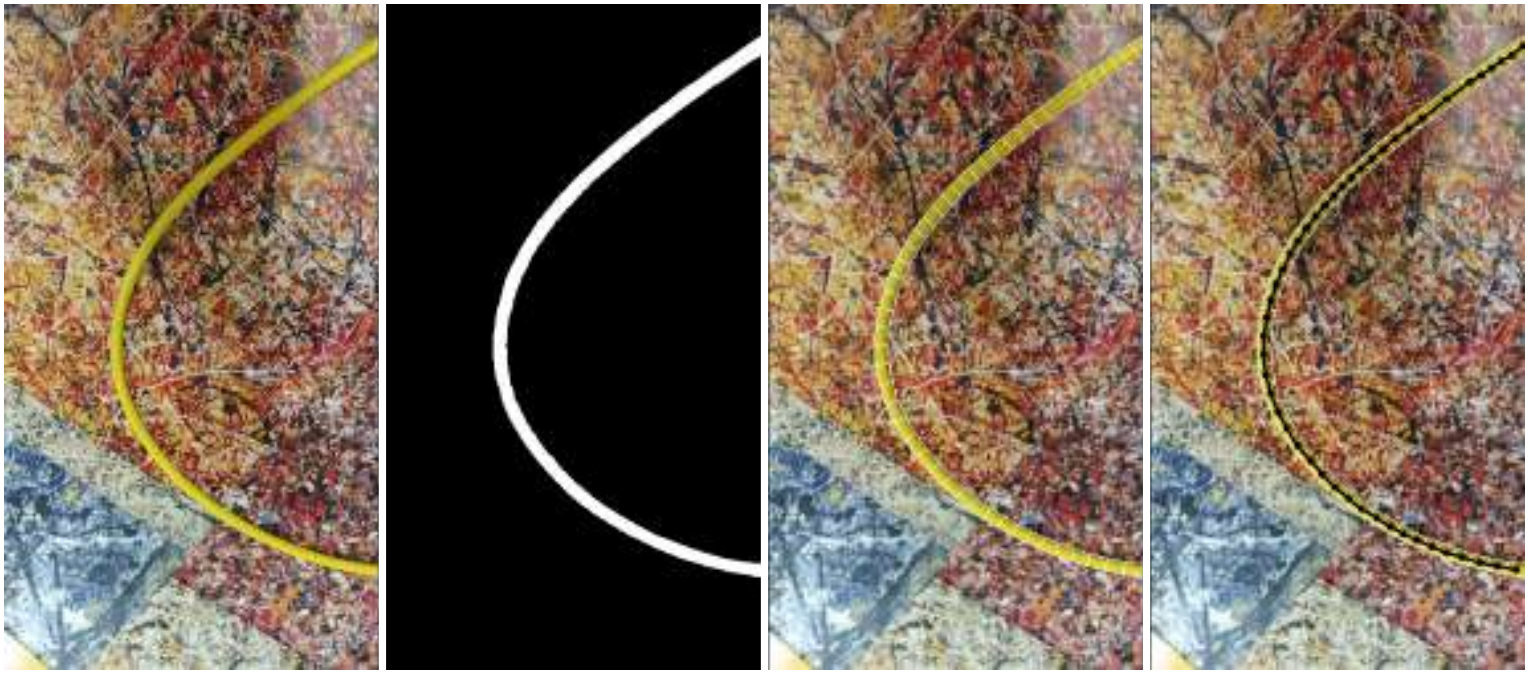
- Superpixellization of the entire image
- Creation of a huge graph
- computations highly time consuming
- results often insufficiently reliable
- manual tuning of a large set of parameters



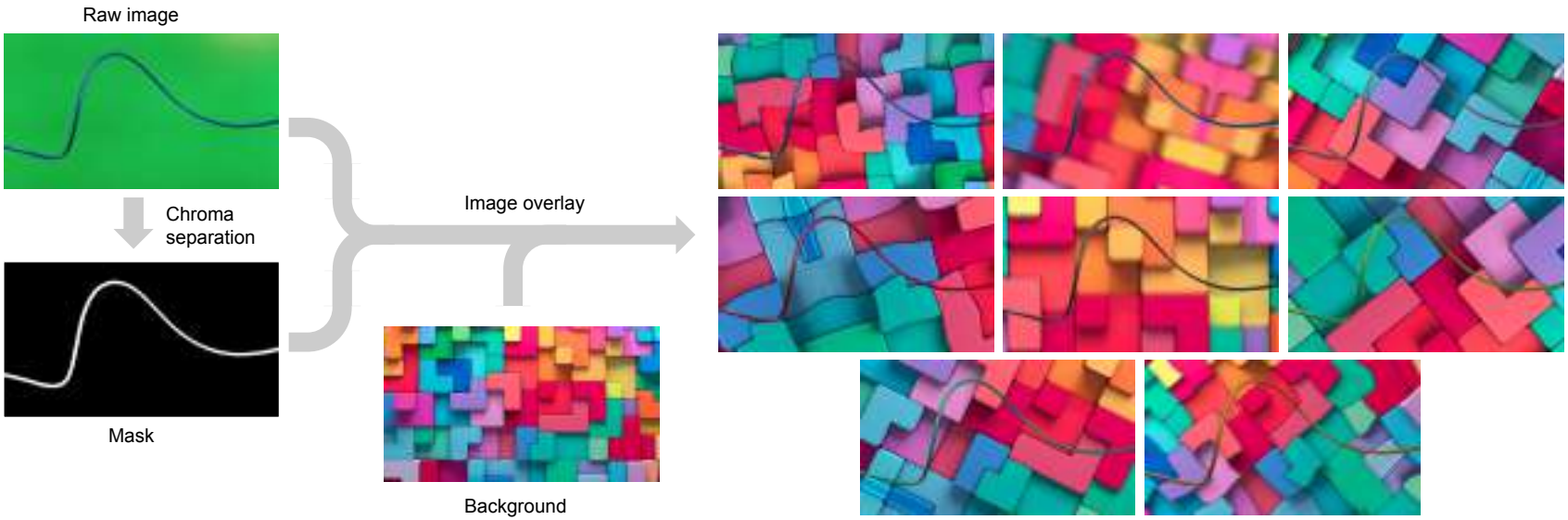
DLO Shape Detection

Ariadne+ [Caporali et al., submitted to IEEE-TII 2021] divides:

- Image segmentation
- Path searching

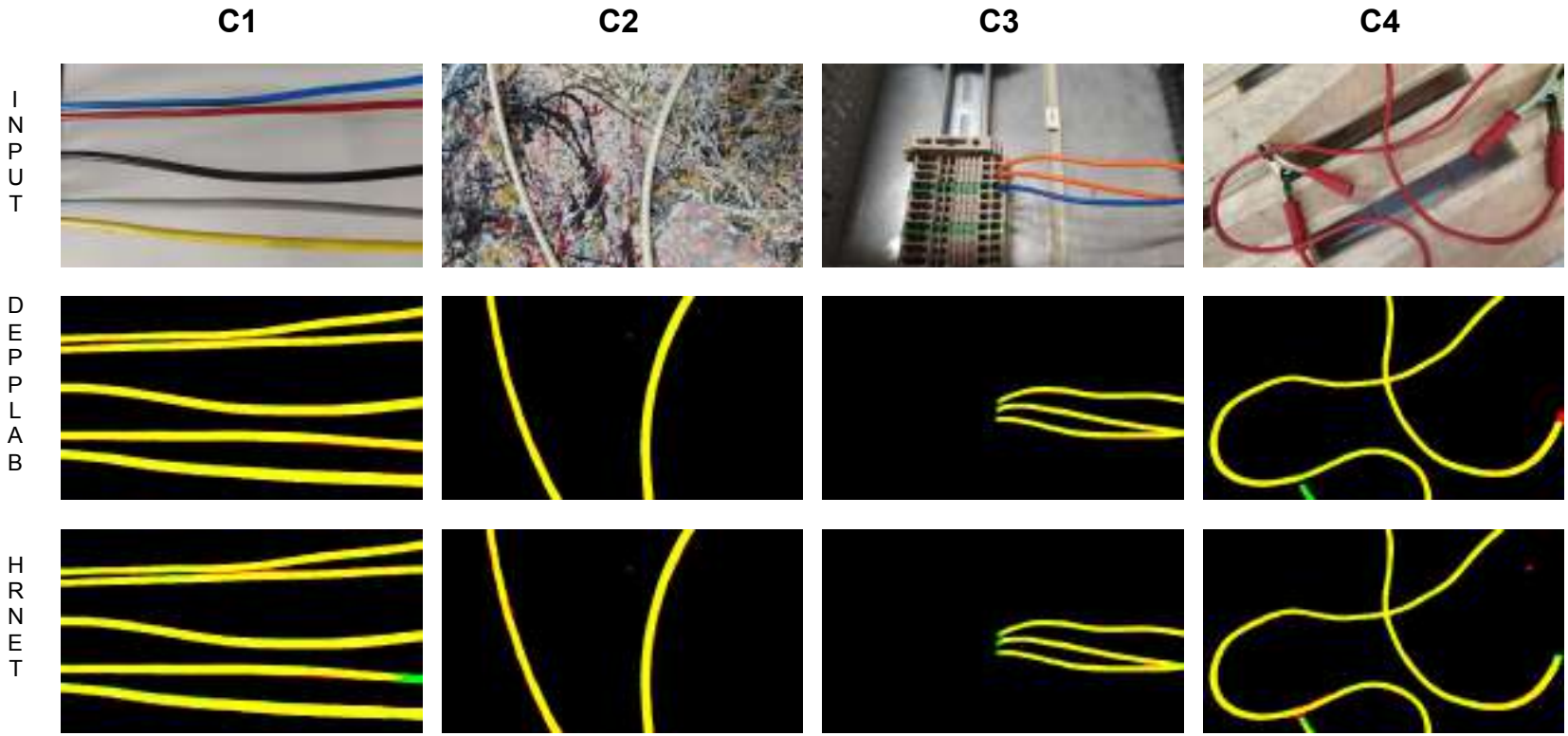


Automatic Labeling

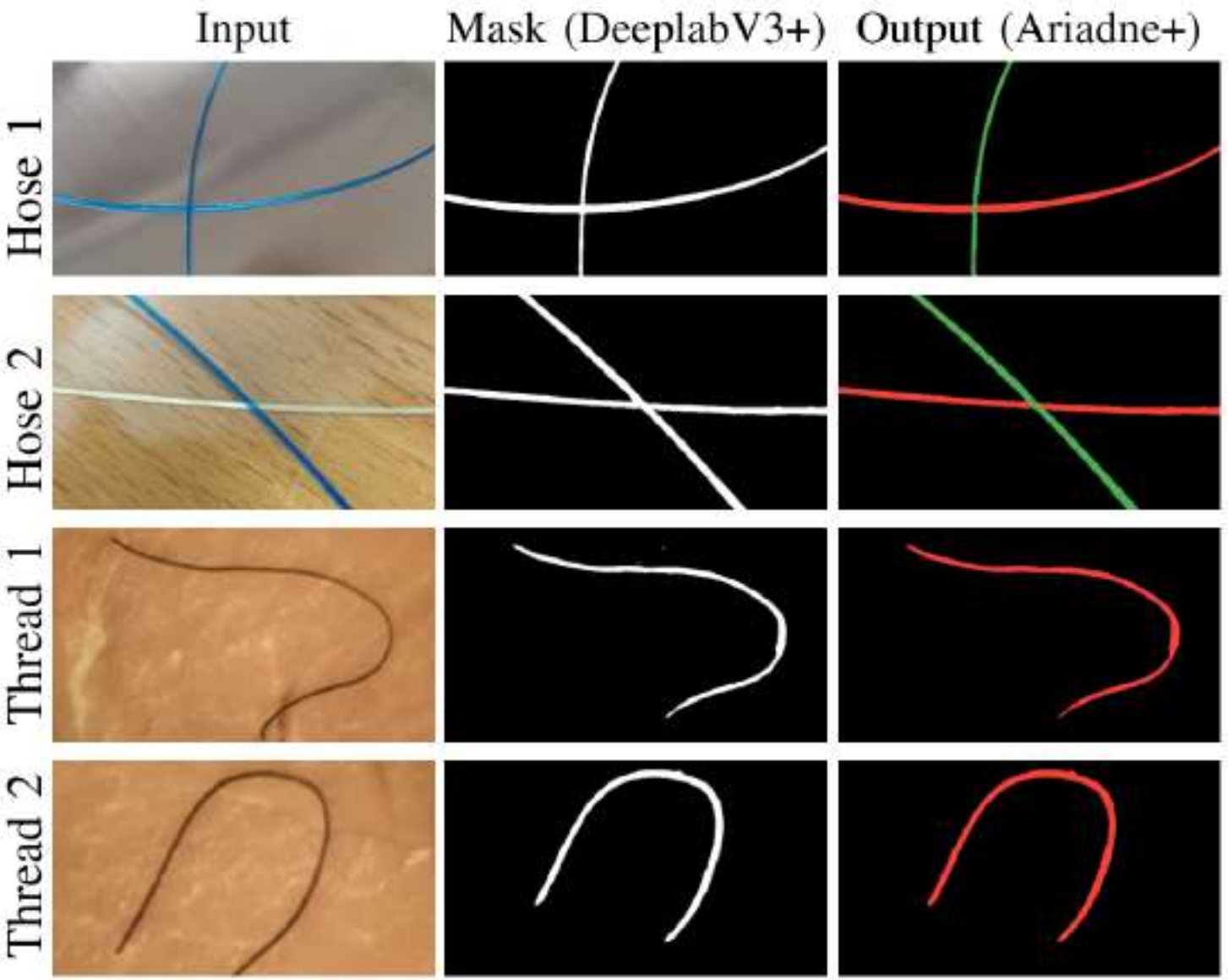


In each new image, foreground and background are separately augmented (by using the mask) before the merging [Zanella et al., ICCCR 2021]

Qualitative Results



Qualitative Results



Dynamic Model of DLOs

DLO Spline-based Modelling

The dynamic model of a DLO can be represented by means of B-spline functions wrt a free coordinate u (the position along the DLO lenght) and suitable **coefficients** or **control points** q_i

$$q(u) = \sum_{i=1}^{n_u} b_i(u) q_i$$

where $q(u) = (x(u), y(u), z(u), \theta(u)) = (r(u), \theta(u))$ is the 4-th dimensional configuration functional space of the cable [Theetten et al., CAD 2007]

From this model, the computation of the shape spatial derivatives is straightforward

$$q'(u) = \sum_{i=1}^{n_u} b_i'(u) q_i$$

$$q''(u) = \sum_{i=1}^{n_u} b_i''(u) q_i$$

Splines Basics

A B-spline (Basis spline) $b_{i,p}(u)$ is a piecewise polynomial function of degree p in a variable u. It is defined over $n_u + 1$ non-descending locations t_i, $t_i \leq t_{i+1}$, called **knots** or breakpoints

$$b_{i,0}(u) := \begin{cases} 1 & \text{if} \quad t_i \leq u < t_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$b_{i,k}(u) := \frac{u - t_i}{t_{i+k} - t_i} b_{i,k-1}(u) + \frac{t_{i+k+1} - u}{t_{i+k+1} - t_{i+1}} b_{i+1,k-1}(u)$$

The B-spline contributes only in the range between the first and last of these knots and is zero elsewhere

Let us define the knot average as

$$\bar{t}_{i,p} = \bar{t}_i = \frac{t_{i+1} + \cdots + t_{i+p}}{p}$$

so the i-th vertex of the spline **control polygon** is (t_i, q_i)

DLO Dynamic Model

The dynamic model of the DLO can be defined as a function of the control points q_i by referring to the Lagrange equations of the system

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_i}\right) = F_i - \frac{\partial U}{\partial q_i}, \quad \forall i \in \{1, \ldots, n_u\}$$

where L is the length of the DLO, F_i is the resultant external force acting on the i-th control point, T is the overall kinetic energy of the system and U is the overall potential energy due to gravity, stretching, bending and torsional effects acting on the DLO.

Including environmental constratints into the dynamic equations

$$\left.\begin{array}{l} \frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_i}\right) = F_i - \frac{\partial U}{\partial q_i} - \sum_{j=1}^{n_c} \lambda_j L_{ji} \\ \Phi(q, \dot{q}) = 0 \end{array}\right\} \forall i \in \{1, \ldots, n_u\}, \; L_{ji} = \frac{\partial \Phi_j}{\partial \dot{q}_i}$$

where λ_j is the Lagrange multiplier relative to the j-th constraint and n_c is the number of constraints.

DLO Kinetic Energy

The kinetic energy of the DLO is due to translation of the control points and rotation of the cross sections. The overall kinetic energy can be represented as a function of the control points q_i as

$$T = \frac{1}{2}\int_0^L \frac{dq^T}{dt} J \frac{dq}{dt} ds, \quad J = \begin{bmatrix} \mu & 0 & 0 & 0 \\ 0 & \mu & 0 & 0 \\ 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & I \end{bmatrix}$$

where $ds = \|r'(u)\|du$ is the element displacement, J is the generalized density matrix of the DLO, μ is the linear density and I is the polar moment of inertia.

Therefore, it is possible to write

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_i}\right) = \sum_{j=1}^{n_u} J \int_0^L b_i(s) b_j(s) ds \, \frac{d^2 q_j}{dt^2}$$

DLO Inertial Force

By considering that $\frac{d^2 q_j}{dt^2} = A_j$ is the acceleration of the j-th control point, the term $J \int_0^L b_i(s) b_j(s) ds = M_{ij}$ can be considered the corresponding inertia term.

Therefore, it is possible to write

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{q}_i}\right) = \sum_{j=1}^{n_u} M_{ij} \ddot{q}_j$$

and, extending this definition to the whole system, this allows to write the overall DLO inertial forces as $M\ddot{q}$, where M is the DLO inertia matrix and $\ddot{q}$ is the vector of the control point accelerations.

DLO Potential Energy

The potential energy U is composed by the gravity effect and by the DLO strain energy due to stretching, bending and torsion.

By defining the strain vector $\epsilon = [\epsilon_s,\ \epsilon_b,\ \epsilon_t]$ including the stretching term ϵ_s, the bending term ϵ_b and the torsional term ϵ_t it follows that the strain energy can be written as

$$U = \frac{1}{2}\int_0^L (\epsilon - \epsilon_0)^T H(\epsilon - \epsilon_0) ds = \frac{1}{2}\int_0^L \epsilon_e^T H \epsilon_e ds$$

where

$$H = \frac{D^2\pi}{4}\begin{bmatrix} E & 0 & 0 \\ 0 & \frac{GD^2}{8} & 0 \\ 0 & 0 & \frac{ED^2}{16} \end{bmatrix}$$

ϵ_0 is the plastic strain, or *strain memory*, of the DLO, that allows to take into account the plasticity of the material, and $\epsilon_e = \epsilon - \epsilon_0$ is the residual strain.

DLO Elastic Forces

The dynamics of the plastic strain can be represented as

$$\dot{\epsilon}_0 = \begin{cases} k_\epsilon(\epsilon_e - \epsilon_M), & \epsilon_e > \epsilon_M \\ 0, & -\epsilon_M \leq \epsilon_e \leq \epsilon_M \\ k_\epsilon(\epsilon_e + \epsilon_M), & \epsilon_e < -\epsilon_M \end{cases}$$

where ϵ_M is a proper maximum strain threshold that generates a *memorization* of the actual strain when it is exceeded and k_ϵ is positive a parameter used to adjust how fast the memorization process evolves.

The elastic forces appearing in the dynamic model can be written as

$$P_i = -\frac{\partial U}{\partial q_i} = -\frac{1}{2}\int_0^L \frac{\partial \epsilon_e^T H \epsilon_e}{\partial q_i} ds = -\int_0^L \frac{\partial \epsilon}{\partial q_i}^T H \epsilon_e ds$$

representing the elastic forces due to the DLO deformation.

DLO Dynamic Model

By exploiting the control point dynamics, it is possible to write the overall DLO dynamic model as

$$M\ddot{q} + N\dot{q} = F + P$$

where the term $N\dot{q}$ is introduced in order to take into account for DLO internal energy dissipation, F is the vector of all the external forces, gravity included, and P is the vector of all the elastic forces.

By including the effect of constraints, the model can be rewritten with n_c additional equations

$$\begin{bmatrix} M & L^T \\ L & 0 \end{bmatrix} \begin{bmatrix} \ddot{q} \\ -\lambda \end{bmatrix} = \begin{bmatrix} F + P - N\dot{q} \\ E \end{bmatrix}$$

where L is the $n_c \times n$ constraint matrix and E encodes the desired constraints behavior.

DLO Time Parametrization

In practical applications, a suitable trade-off between precision and computation time is needed. For this reason, a time-based bivariate representation of the DLO configuration is introduced

$$q(u,t) = \sum_{i=1}^{n_u} \sum_{j=1}^{n_t} b_i(u) b_j(t) q_{ij} \;\rightarrow\; q(t) = B_t(t) q_t$$

where q_{ij} is a $n_u \times n_t$ set of control points used to parameterize the problem. The time derivative of the DLO configuration is straightforward

$$\dot{q}(u,t) = \sum_{i=1}^{n_u} \sum_{j=1}^{n_t} b_i(u) b_j'(t) q_{ij} \;\rightarrow\; \dot{q}(t) = B_t'(t) q_t$$

$$\ddot{q}(u,t) = \sum_{i=1}^{n_u} \sum_{j=1}^{n_t} b_i(u) b_j''(t) q_{ij} \;\rightarrow\; \ddot{q}(t) = B_t''(t) q_t$$

This allows to significantly reduce the number of points to be computed in order to compute the DLO dynamics [Palli, ICPS2020]

DLO Time Parametrization

Therefore, the DLO dynamic model can be simply rewritten as

$$MB_t'' q_t + NB_t' q_t = F + P$$

where the dependence from time of B_t and its derivatives is omitted for brevity. The left-hand side is clearly linear in q_t, but since P depends on q, i.e. $P = P(q)$, the overall system is nonlinear and can not be solved efficiently in this form

The DLO dynamic model can be solved in a discrete-time fashion by evaluating the solution of the system at time t_{k+1} given the solution at time t_k, i.e.

$$MB_t'' q_{k+1} + NB_t' q_{k+1} = F + P(B_t q_{k+1})$$

DLO Model Integration

However, a first step toward the simplification of this model is to rewrite the elastic energy term $P(q_{k+1})$ in a linearized way

$$MB''q_{k+1} + NB'q_{k+1} + KB_t\Delta q_t = F + P(B_t q_k)$$

It follows that the linearized model of the DLO can be written as

$$MB_t''q_{k+1} + NB_t'q_{k+1} + KB_t q_{k+1} = F + P(B_t q_k) + KB_t q_k$$

where now a clear separation between the values of q_t at time t_k and t_{k+1} is obtained

Note that this model can be also used to find the DLO equilibrium by removing the terms $MB_t'' + NB_t'$

DLO Model Integration

In particular, by collecting q_{k+1} the following linear system is obtained

$$(MB_t'' + NB_t' + KB_t)q_{k+1} = F + P(B_t q_k) + KB_t q_k$$

or, in a more compact way

$$q_{k+1} = \Lambda^{-1}\Psi(q_k) + \Lambda^{-1}F$$

$$\Lambda = MB_t'' + NB_t' + KB_t$$

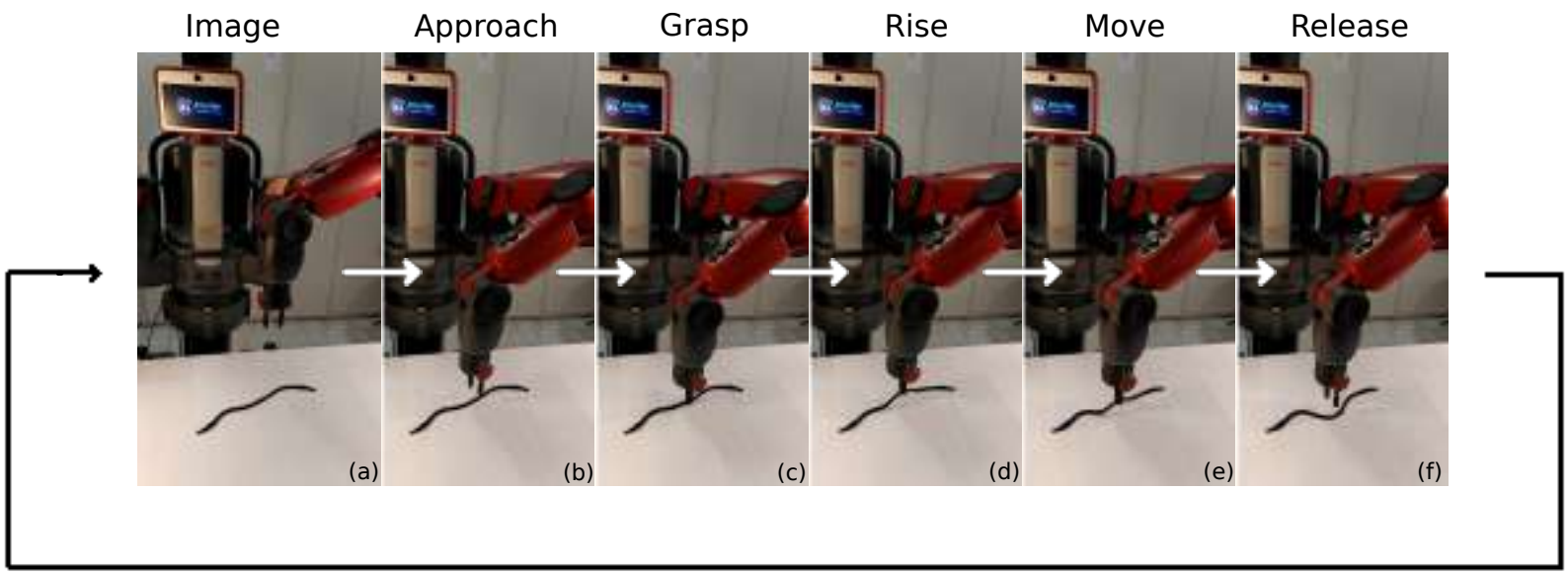
$$\Psi(q_k) = P(B_t q_k) + KB_t q_k$$

where Λ is a symmetric positive definite regression matrix, i.e. it is always invertible, assuming that the time-related spline basis B_t is properly defined

Optimization of the DLO Manipulation Task

DLO Manipulation Strategy

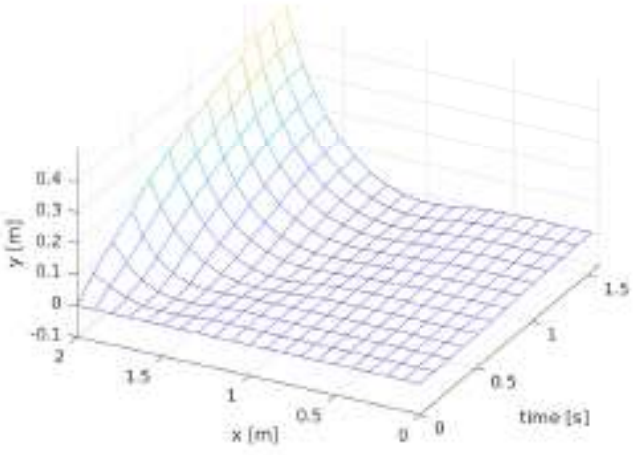
Definition of the manipulation primitive



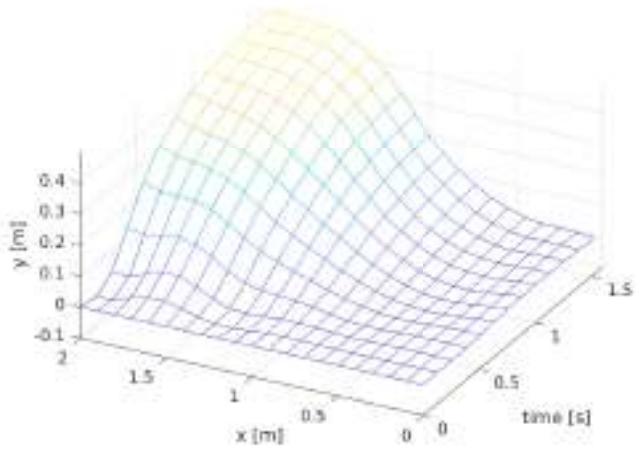
Alternative approaches:

- Optimal manipulation sequence based on the DLO model
- A decision-making process based on deep Q-learning

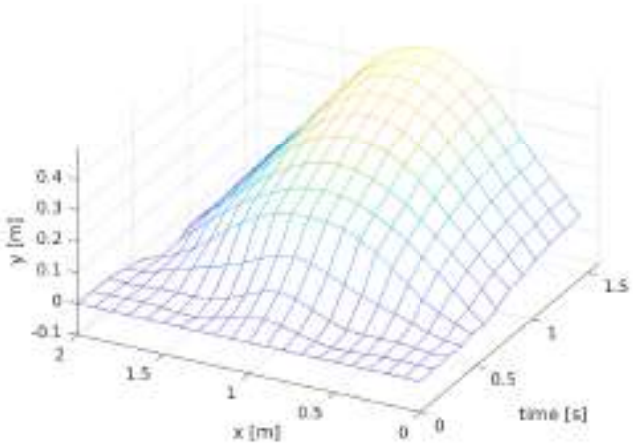
DLO Manipulation Parametrization



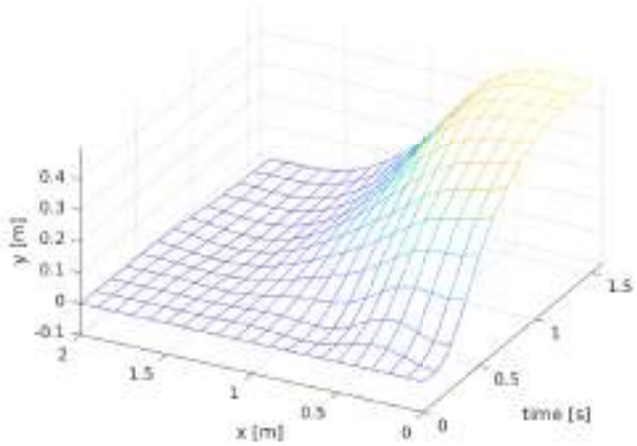
(a) DLO grasped at the distal end.



(b) DLO grasped at 0.5 m from distal end.

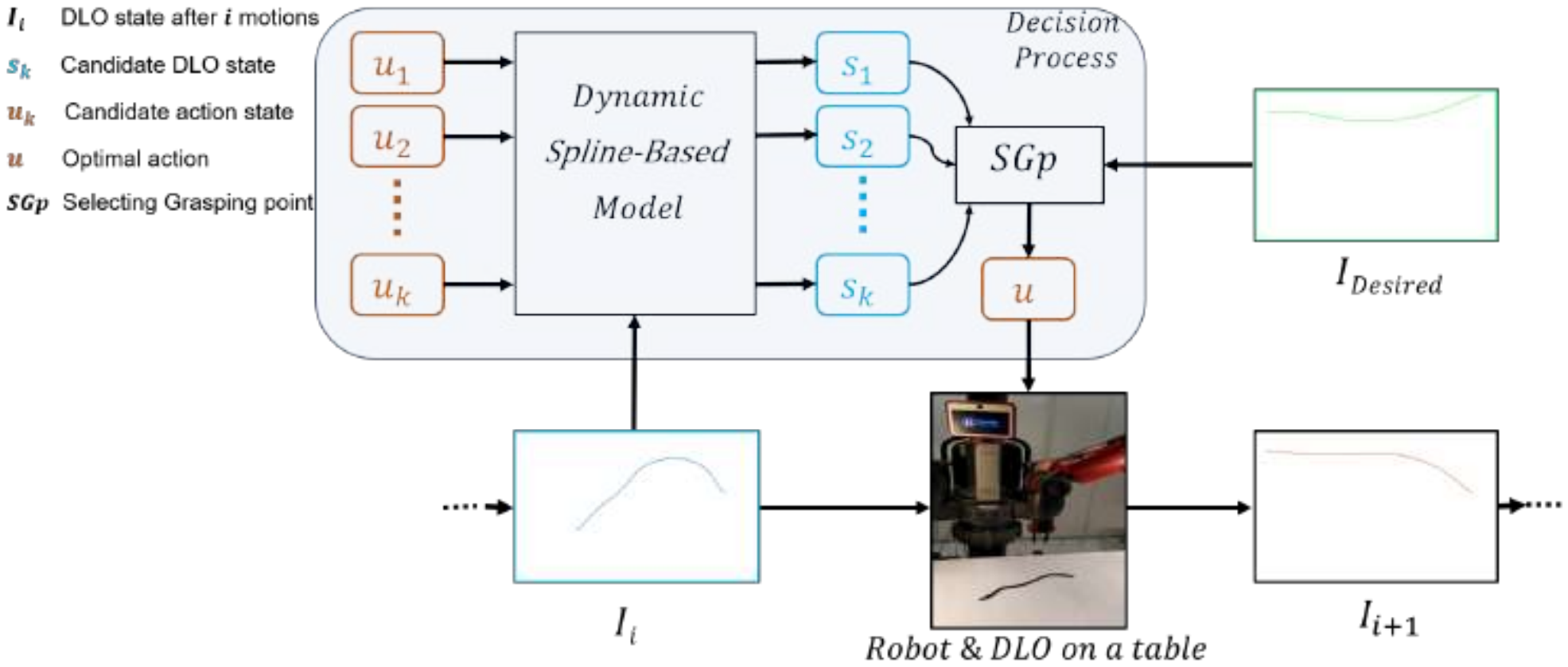


(c) DLO grasped at the middle point.



(d) DLO grasped 0.5 m from proximal end.

DLO Manipulation Strategy



Optimal DLO Manipulation

Given the actual DLO shape, the problem consists on selecting the best point to minimize the distance between the DLO shape q_0 measured by the vision system and the target shape q_g

$$d = \|q_0 - q_g\|$$

The grasping point is parametrized by $v \in [0,\ L]$, the same domain of u, so it is possible to write $d = d(v)$

By supposing suitable smoothness in the variation of d with respect to v, it is possible to express $d(v)$ by a suitable spline interpolation

$$d(v) = \sum_{h=1}^{n_v} b_h(v) d_{v_h} = B_v(v) d_v$$

where $b_h(v)$ is the h-th element of the spline basis used to interpolate $d(v)$ and $d_{v_h},\ h \in [1, \cdots, n_v]$ is the set of n_v coefficients, or control points, of the interpolation, $B_v(v)$ is the line vector collecting all the $b_h(v)$ values and d_v is the vector of interpolation coefficients

Parametrization of the Manipulation Primitive

The control points d_v can be computed by selecting a suitable set of collocation points for v, let us call then $v_l, l \in [1, \cdots, n_l]$, in such a way that the linear system

$$d_l = B_l d_v, \quad d_l = \begin{bmatrix} d(v_1) \\ \vdots \\ d(v_{n_v}) \end{bmatrix}, \quad B_l = \begin{bmatrix} B_v(v_1) \\ \vdots \\ B_v(v_{n_v}) \end{bmatrix}$$

is provided with a full-rank matrix B_l. It follows that

$$d_v = B_l^+ d_l$$

provides the control points needed to describe how the shape distance change with the grasping point after each manipulation primitive

Optimal Manipulation

The optimal grasping point can be found over the spline representation provided by d_v by solving the following minimization problem [Khalifa and Palli, submitted to JAMT, 2021]

$$v^* = \min_v \{v \in [0,\ L] : \|B_v(v)d_v\|\}$$

Since $d(v)$ is clamped curves, if the minimum is at an extremum, the control point at the corresponding extremum is the minimum

$$v^* = \begin{cases} 0, & B'_v(v)d_v < 0\ \forall v \in [0,\ L] \\ L, & B'_v(v)d_v > 0\ \forall v \in [0,\ L] \end{cases}$$

Otherwise, if an internal minimum exists, it must be within the convex hull of a set of 4 control points in which the intermediate 2 are the first negative and the second positive. This means that v^* can be computed efficiently

Spline Properties

First, to find the zeros of $d(v)$, its derivative is considered

$$d'(v) = B'_v(v)d_v$$

which control points can be defined according to the original spline formulation as

$$d'_v = B_l^+ B'_v d_v$$

Therefore, to find the zeros of $d'(v)$

- A spline whose control polygon is everywhere of one sign cannot have any zeros
- A good starting point to find zeros is in the neighbourhood of a zero of the control polygon
- The control polygon converges to the spline as the spacing of the knots goes to zero

So the basic idea is to add knots close to the control polygon zeros until a minimum knot distance is obtained

Spline Zeros Finding Algorithm

The first zero of the control polygon is the zero of the linear segment connecting the two points $(t_{k-1}, d'_{v_{k-1}})$ and (t_k, d'_{v_k}), where k is the smallest integer such that $d'_{v_{k-1}} d'_{v_k} \leq 0$ and $d'_{v_{k-1}} \neq 0$ ($d'_{v_{k-1}} < 0$ and $d'_{v_k} \geq 0$ in our case)

The zero is characterised by the equation

$$(1 - \lambda) d'_{v_{k-1}} + \lambda d'_{v_k} = 0$$

which has the solution

$$\lambda = \frac{-d'_{v_{k-1}}}{d'_{v_k} - d'_{v_{k-1}}}$$

Therefore, the control polygon has a zero at

$$u^* = (1 - \lambda)\bar{t}_{k-1} + \lambda \bar{t}_k = \bar{t}_k - \frac{d'_{v_k}(t_{k+p} - t_k)}{p(d'_{v_k} - d'_{v_{k-1}})} = \bar{t}_k - \frac{d'_{v_k}}{\Delta d'_{v_k}}$$

from which it is apparent that this is a discrete version of Newton's method

Spline Zeros Finding Algorithm

If we insert a new knot u^* in $\mathbf{t}$ and form the new knot vector $\mathbf{t}^1 = \mathbf{t} \cup \{u^*\}$ we obtain $d'^1_{v_i} = d'_{v_i}$ for $i = 1, \ldots, k-1-p$,

$$d'^1_{v_i} = (1-\mu_i) d'_{v_{i-1}} + \mu_i d'_{v_i}, \;\; \mu_i = \frac{u^* - t_i}{t_{i-p} - t_i}, \;\; i = k-p, \ldots, k-1$$

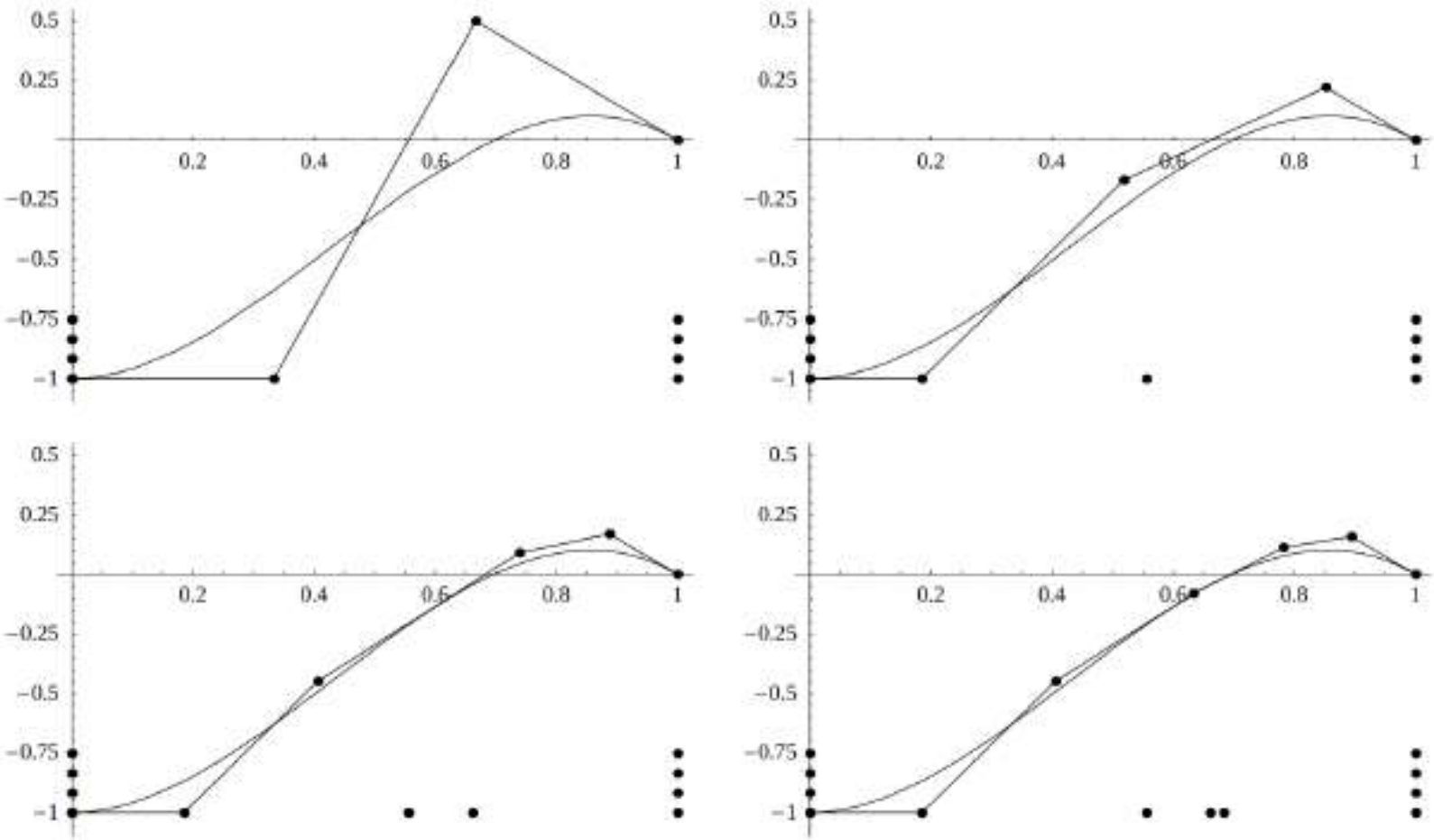
and $d'^1_{v_i} = d'_{v_{i-1}}$ for $i = k, \ldots, n+1$. It is not hard to verify that the same relation holds for the knot averages

$$\bar{t}^1_i = (1-\mu_i)\bar{t}_{i-1} + \mu_i \bar{t}_i, \;\; i = k-p, \ldots, k-1$$

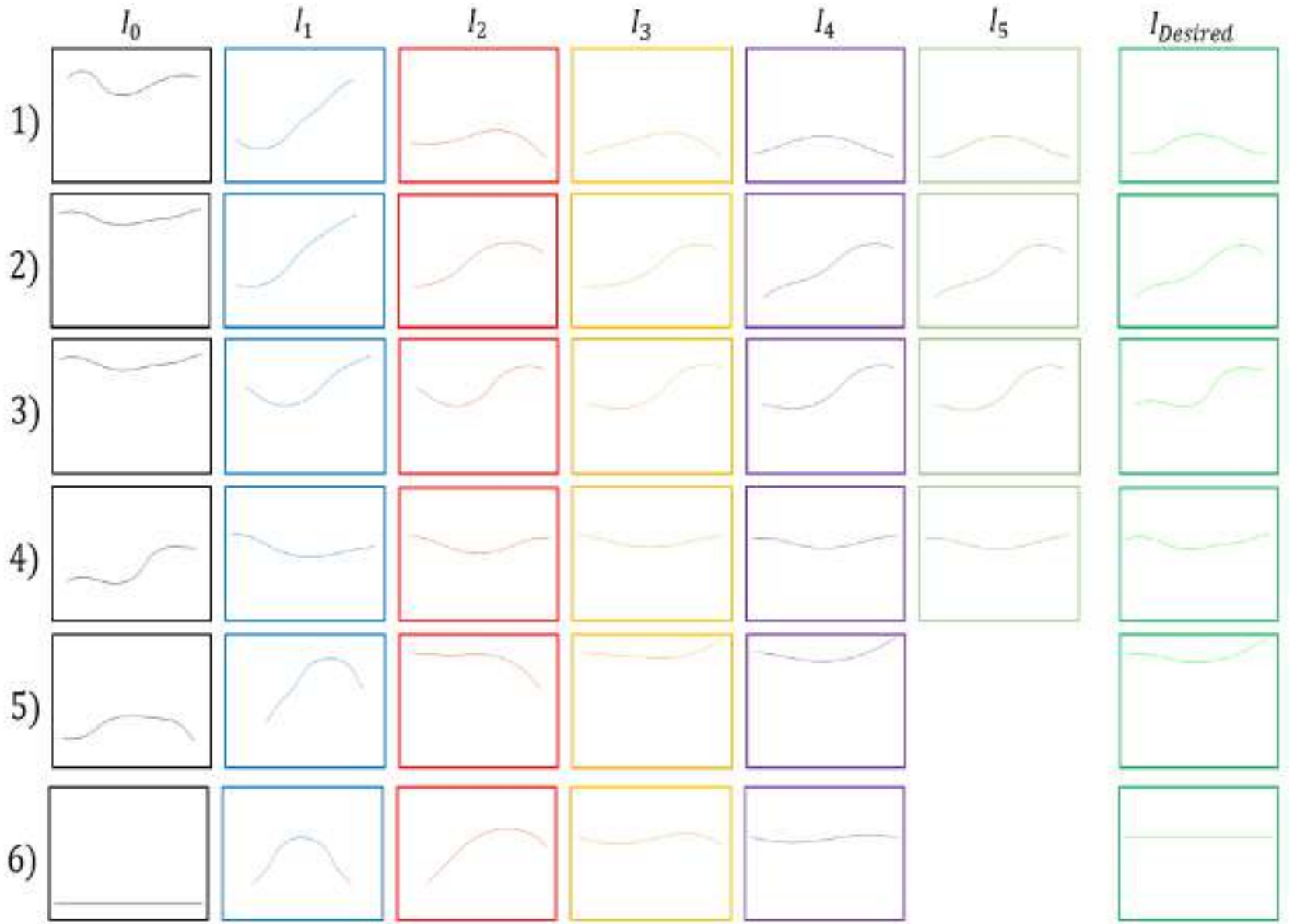
This means that the corners of the refined control polygon lie on prevous ones.

The procedure can be iterated until a minimum distance among the knots is obtained or no adjacent coefficients with opposite sign can be found

Spline Zeros



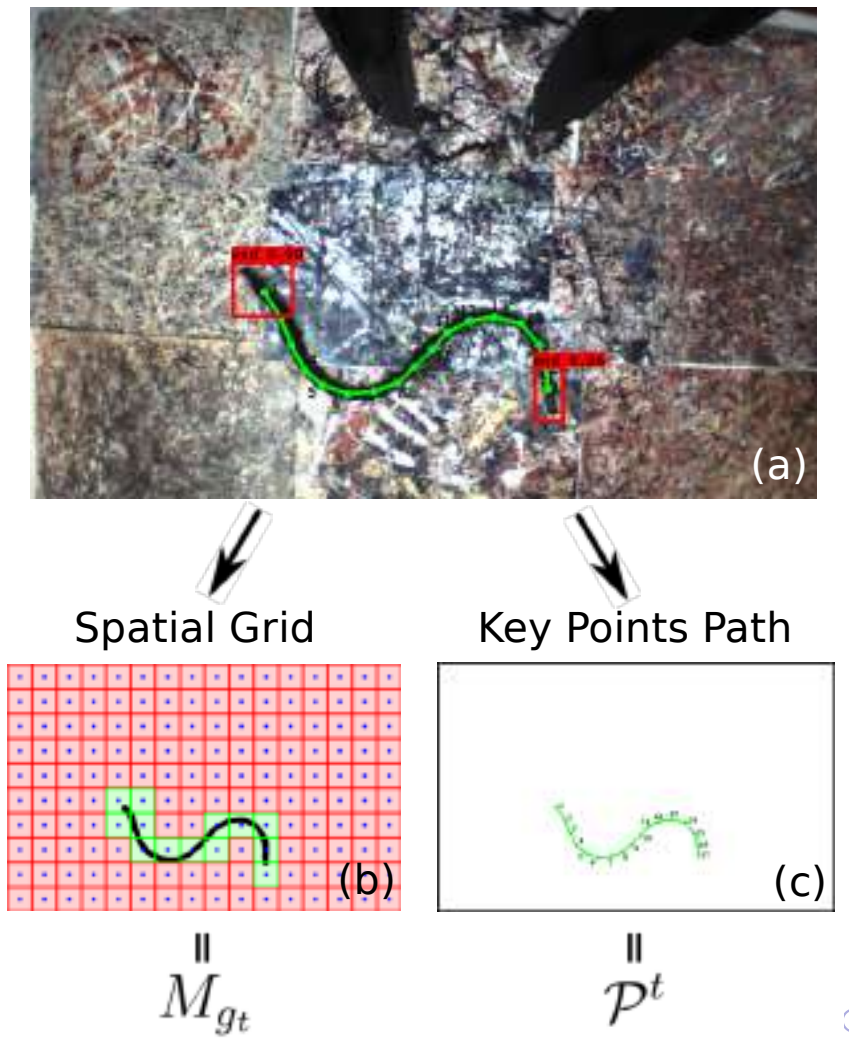
DLO Manipulation Results



Spatial Grid Model

A uniform space partitioning is performed on the binary mask [Zanella and Palli, submitted to INS, 2021]

Each region is mapped into a scalar value $g_t^{i,j} = \Omega_{[0,1]}\left(\frac{1}{\psi_h \psi_w} \sum_{u,v \in \Psi_{i,j}} I_t^{\text{mask}}[u,v], g^{\text{Th}}\right)$ that is the average of all the region-pixels binarized through the function $\Omega_{[0,1]}(x, x^{\text{Th}})$, which gets 1 only when $x \geq x^{\text{Th}}$ and 0 otherwise. From these values we define the spatial grid matrix at time t as $M_{g_t} = [g_t^{i,j}]_{i \in n_{\text{rows}}, j \in n_{\text{cols}}} \in [0,1]^{n_{\text{rows}} \times n_{\text{cols}}}$, where every cell (i,j) and every region $\Psi_{i,j}$ have a bijective correspondence



Decision Process

The state s_t is a linear combination of the spatial grid matrix of the scene M_{g_t} and the target one M_{g^*}

$$s_t = 2M_{g_t} + M_{g^*}.$$

In this way the state is a matrix $s_t \in [0,3]^{n_{\text{rows}} \times n_{\text{cols}}}$ where each element $s_t^{i,j}$ corresponds to the cell (i,j) of the spatial grid built on the scene. Note that it can be rewritten as

$$s_t^{i,j} = \begin{cases} 0 \text{ if } \Psi_{i,j} \text{ is part of the background} \\ 1 \text{ if } \Psi_{i,j} \text{ is part of the target shape only} \\ 2 \text{ if } \Psi_{i,j} \text{ is part of the current shape only} \\ 3 \text{ if } \Psi_{i,j} \text{ is an overlapped region} \end{cases},$$

where the overlapped regions are set of image pixels belonging to both the target and the current shape.

Decision Process

The manipulation task is represented as a Markov decision process defined by $(\mathcal{S}, \mathcal{A}, p, r)$ with state space $\mathcal{S}$ and action space $\mathcal{A}$

Given the current state s_t, the current action a_t is selected according to the policy $\pi(a_t|s_t)$, which implies the transition of the environment to a new state s_{t+1} with unknown probability density $p(s_{t+1}|s_t, a_t)$ and the formulation of a reward r_t

Goal: find π^* maximizing $\sum_{t=i}^{+\infty} \mathbb{E}_{(s_t,a_t)\sim p_\pi}[r_t]$

DQN is used to approximate the Q-value function $Q_\pi(s_t, a_t) = \sum_{t_i=t}^{T} \mathbb{E}_{\pi_\theta}[r_t|s_t, a_t]$ measuring the expected reward of taking action a_t in state s_t at time t

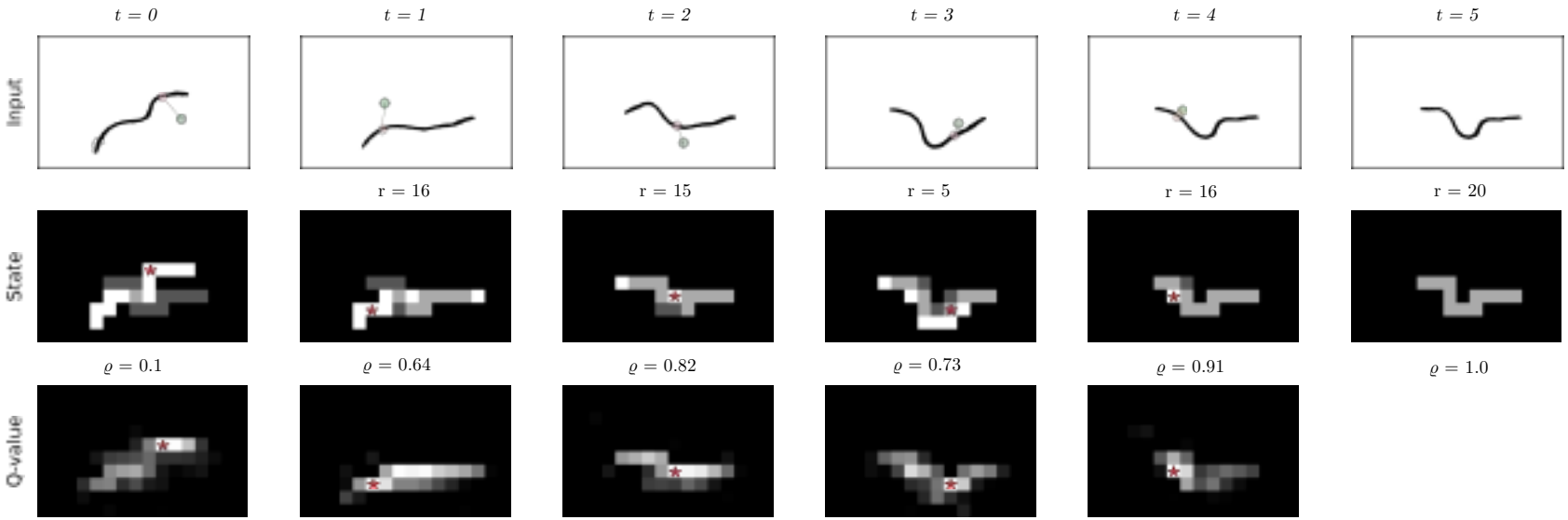
The learning objective is to iteratively minimize the error δ_t

$$\delta_t = |Q_\pi(s_t, a_t) - y_t|$$

$$y_t = r_t + \gamma\, Q_\pi\left(s_{t+1}, \underset{a'\in\mathcal{A}}{\text{argmax}}\, Q_\pi\left(s_{t+1}, a'\right)\right)$$

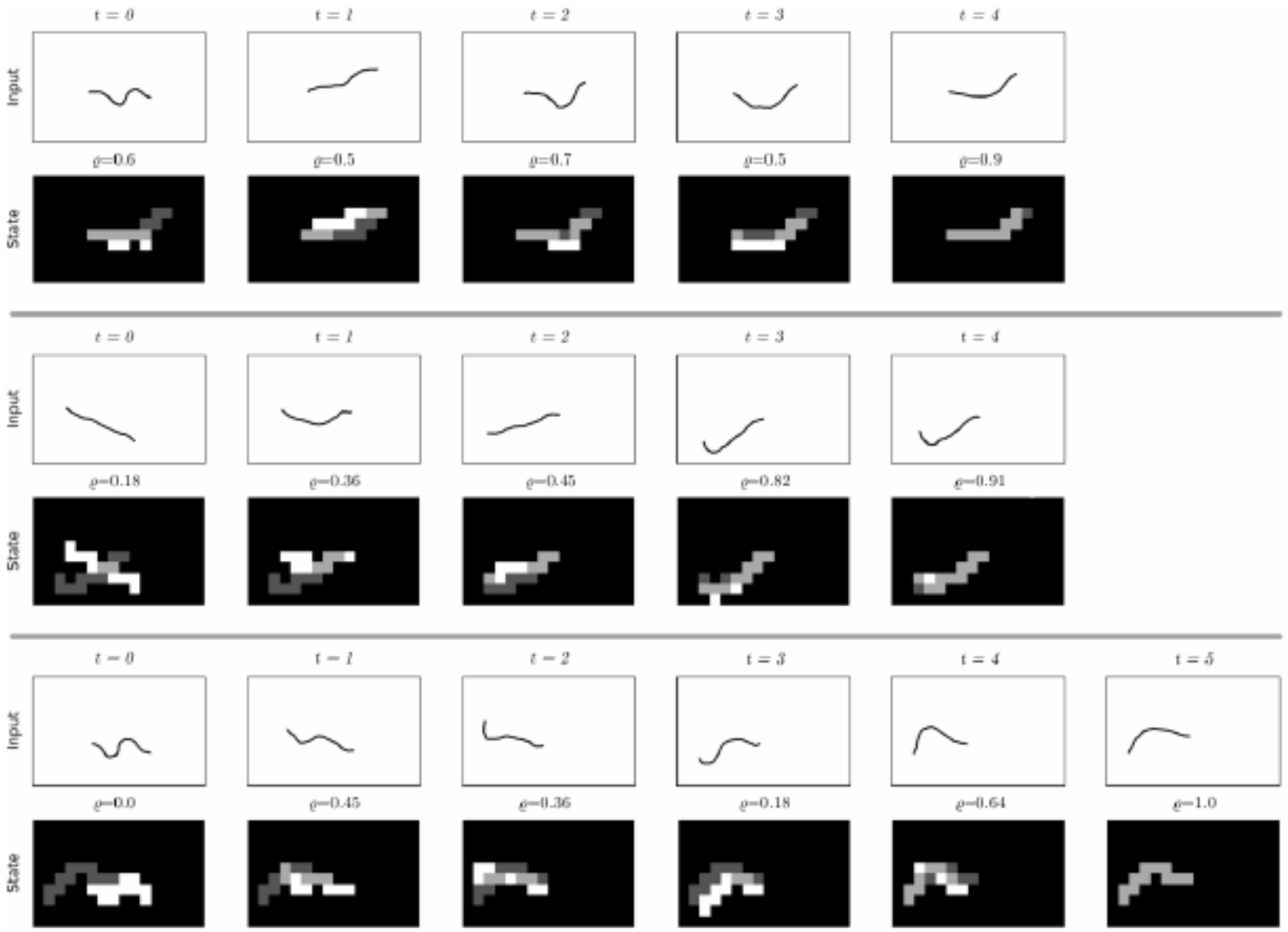
with discount rate $\gamma > 0$

Data-Driven Reshape Sequence



VIDEO

Data-Driven Reshape Sequence: More Samples

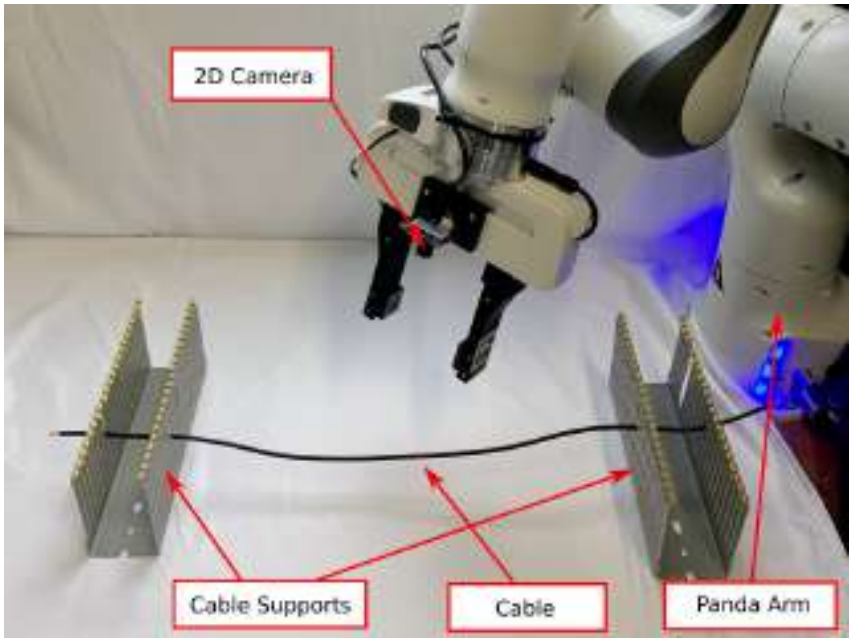


DLO State Perception in 3D

3D DLO Shape Reconstruction

In most of the cases, the 3D shape is needed [Caporali et al., AIM2021]

- A 2D camera is mounted on a robot arm
- The cables are detected using Ariadne+
- The estimated 2D splines obtained from multiple viewpoints are used for the 3D shape estimation



Ariadne+ provides the DLO as a 2D vector $p(u) = [p_x(u)\ p_y(u)]^T$ of pixel coordinates representing the estimated spline samples in the input image

3D DLO Shape Reconstruction

Let us consider the case in which a single unknown point x in the Cartesian space and expressed with respect to the world reference frame is observed by the camera mounted on the robot from multiple points of view.

Provided that the camera frame with respect to world frame at the i-th points of view is

$$ {}^{w}T_{c_i} = \begin{bmatrix} {}^{w}R_{c_i} & {}^{w}t_{c_i} \\ 0\ 0\ 0 & 1 \end{bmatrix} $$

where ${}^{w}R_{c_i}$ is the rotation matrix and ${}^{w}t_{c_i}$ is the position of the camera frame origin in world coordinates, we assume the point x is seen in the image related to the i-th points of view at $p_i = [p_{x_i}\ p_{y_i}]^T$, being p_{x_i} and p_{y_i} the point pixel coordinates in the image.

3D DLO Shape Reconstruction

A so-called unit ray v_i passing through the image reference frame origin and x can be expressed in the image frame considering the pixel coordinates p_i and the camera focal distance f

$$v_i' = \begin{bmatrix} p_{x_i} - c_x \\ p_{y_i} - c_y \\ f \end{bmatrix}, \quad v_i = \frac{v_i'}{\|v_i'\|}$$

where c_x and c_y are the pixel coordinates of the image center (assuming the camera frame is centered with respect to the image). Then, v_i can be expressed in the world frame by

$$^w v_i = {}^w R_{c_i} v_i$$

Provided that n_p distinguished points of view are available, the estimation $\tilde{x}$ of the unknown point x can be obtained by looking for the point having the minimum distance from all the rays.

By defining the symmetric V_i matrix

$$V_i = I - {}^w v_i \, {}^w v_i^T$$

providing the seminorm on the ray distance, the point location estimate $\tilde{x}$ is provided by nearest point search algorithm, i.e.

$$\tilde{x} = \left(\sum_{i=1}^{n_p} V_i \right)^{-1} \left(\sum_{i=1}^{n_p} V_i \, {}^w t_{c_i} \right)$$

We assume the same portion of the DLO is seen in all the images. Let us call the set of spline samples $p_{ij} = p_i(u_j)$, $j = 1, \ldots, n_s$, $i = 1, \ldots, n_p$, where n_s is the number of spline samples, n_p is the number of points of view, $p_i(\cdot)$ is the spline provided for by the i-th image and u_j are the spline sample points.

The vector of control points $q_v = [q_1 \ \cdots \ q_{n_u}]^T$ of the 3D spline $q(u)$ that optimally approximated the set of point estimates p_{ij} can be defined as

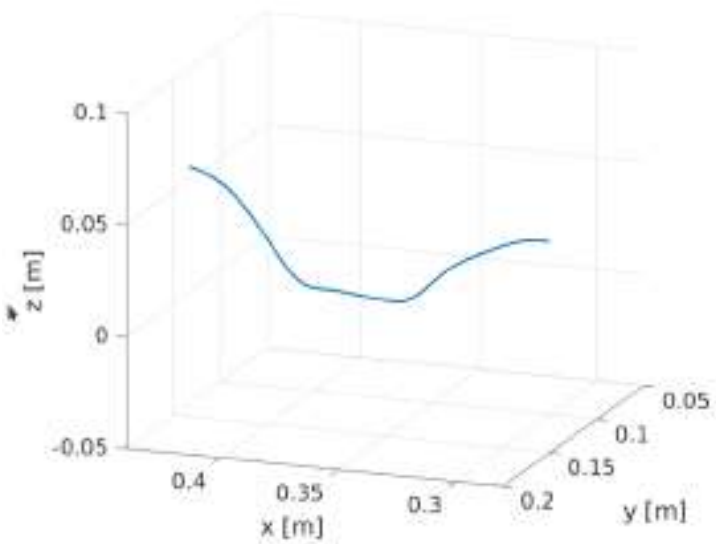
$$q_v = B^{\#} \tilde{x}_v$$

where $^{\#}$ represents the matrix pseudoinverse and

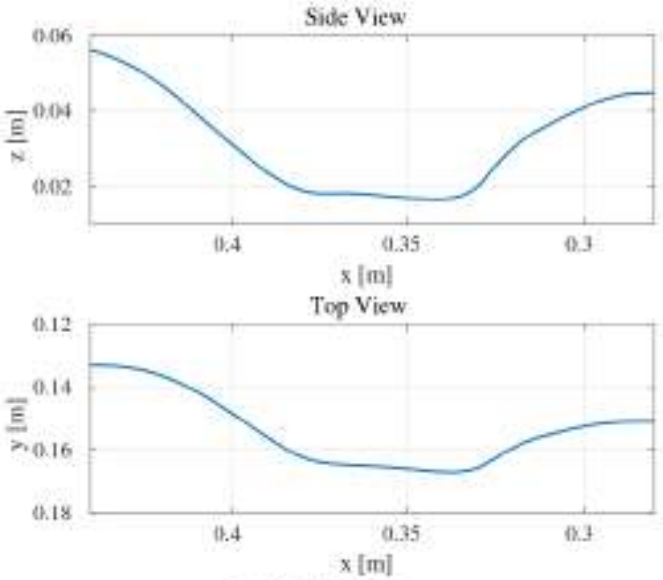
$$B = \begin{bmatrix} b_1(u_1) & \cdots & b_{n_u}(u_1) \\ b_1(u_2) & \cdots & b_{n_u}(u_2) \\ \vdots & \vdots & \vdots \\ b_1(u_{n_s}) & \cdots & b_{n_u}(u_{n_s}) \end{bmatrix}, \ \tilde{x}_v = \begin{bmatrix} \left(\sum_{i=1}^{n_p} V_{i1}\right)^{-1} \left(\sum_{i=1}^{n_p} V_{i1} \, {}^w t_{c_i}\right) \\ \left(\sum_{i=1}^{n_p} V_{i2}\right)^{-1} \left(\sum_{i=1}^{n_p} V_{i2} \, {}^w t_{c_i}\right) \\ \vdots \\ \left(\sum_{i=1}^{n_p} V_{in_s}\right)^{-1} \left(\sum_{i=1}^{n_p} V_{in_s} \, {}^w t_{c_i}\right) \end{bmatrix}$$

being V_{ij} the matrix computed for the j-th sample provided by the i-th image.

3D Shape Reconstruction Results

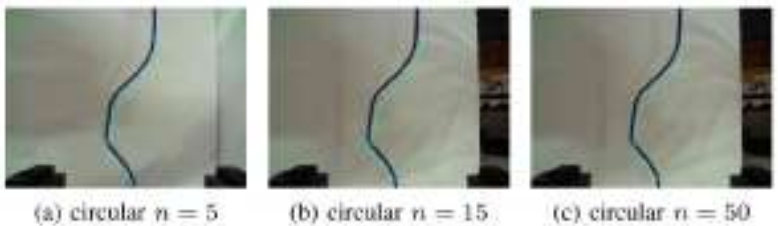


(a) 3D Cable Plot.



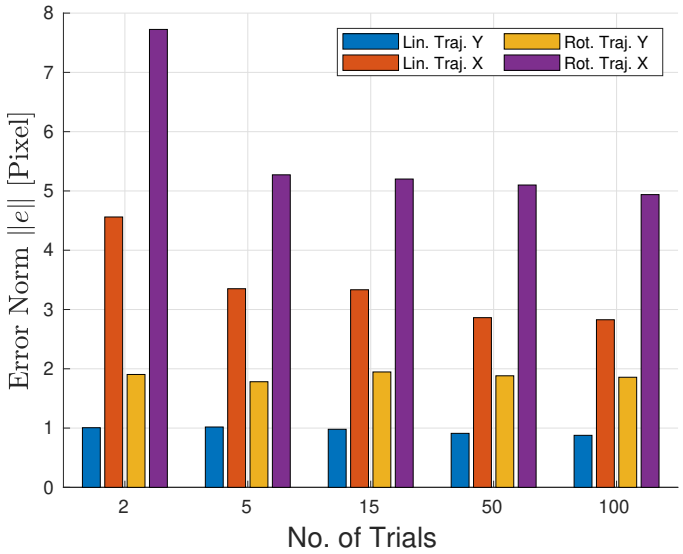
(b) 2D Cable Plots.

(c) Real Cable Shape.

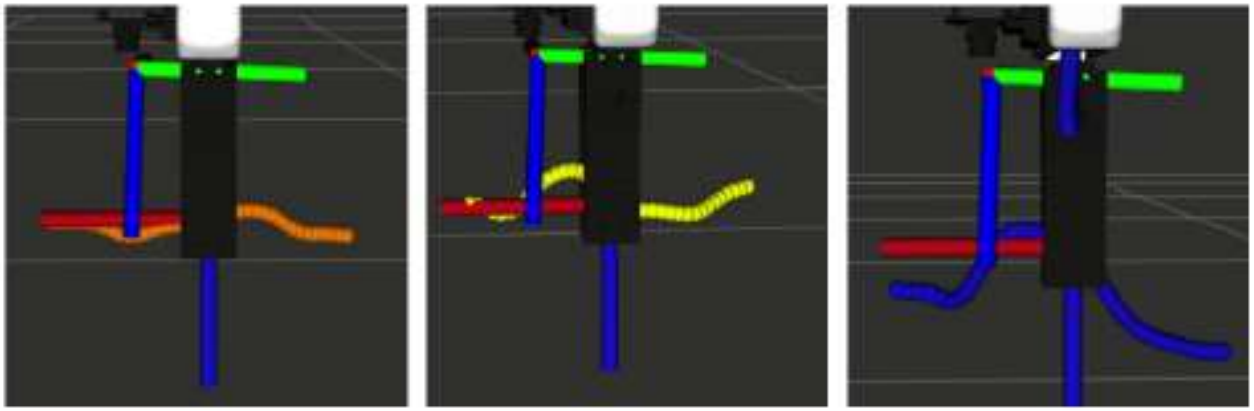


(a) circular $n = 5$ (b) circular $n = 15$ (c) circular $n = 50$

(d) linear $n = 5$ (e) linear $n = 15$ (f) linear $n = 50$



3D Shape Reconstruction Results



(a) Representation of the 3D cable shape estimation in RViz.

(b) $\phi = 2mm$

(c) $\phi = 3mm$

(d) $\phi = 4mm$

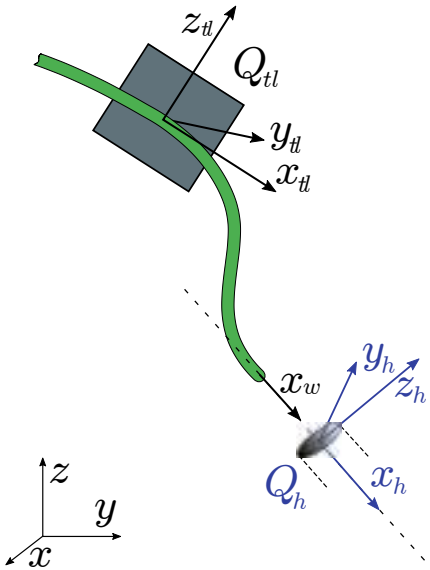
VIDEO

DLO-in-Hole Problem

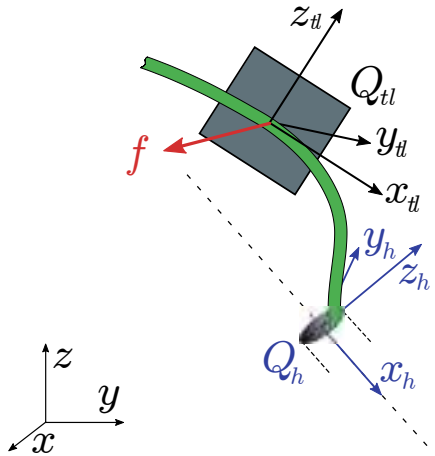
Insertion Task

Problem: Inserting a DLO into an hole [Zanella et al., CoDIT 2019]

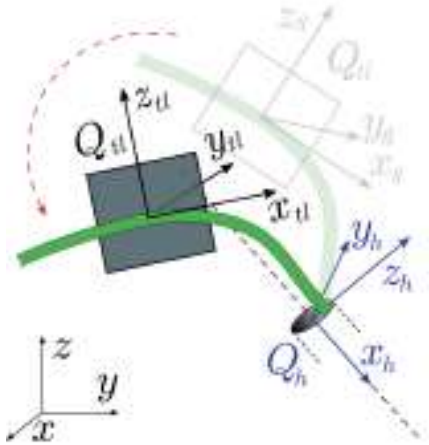
Initial condition



Wire-hole contact



Tool-pose correction



Tactile Feedback

Other Approaches:

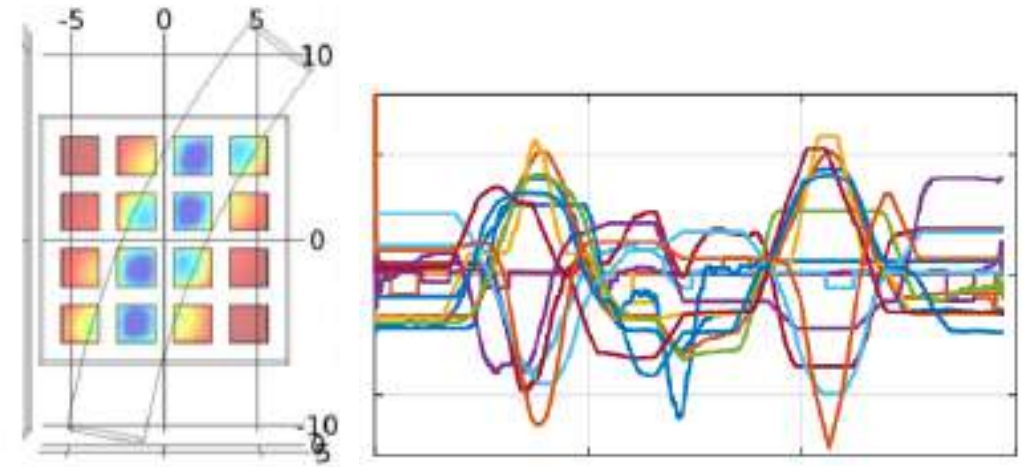
- **Vision systems** are an unfeasible or limited solution in tight spaces;
- **Force/Torque sensors** increase the system price and provide signals affected by undesired effect difficult to manage.

Our Solution:

- **Tactile** data integrating a **RNN** to estimate contact forces.

Tactile Feedback

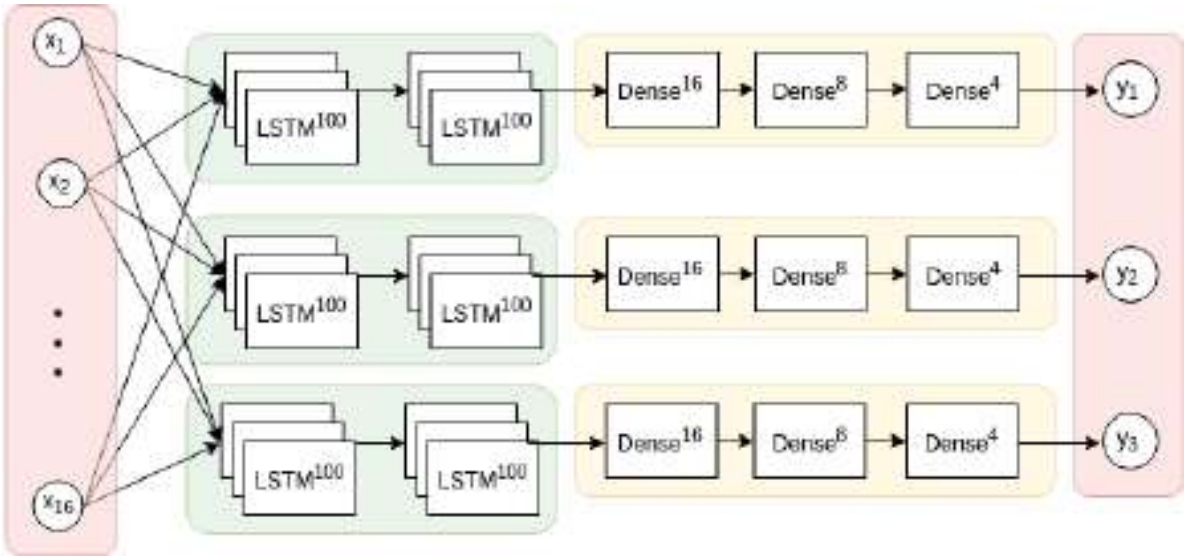
Tactile sensor specifically designed for wires grasping:
16 optoelectronic taxels and a flat deformable layer.



The 16 signals are mapped into a 3-dimensional vector which represents an estimation of the force vector on the fingers.

Tactile Feedback RNN

Three decoupled regressors, for the 3 force components.

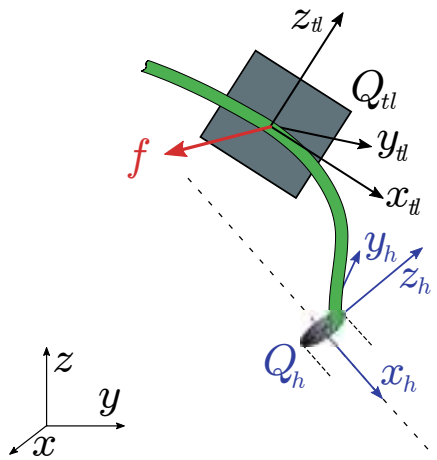


Ground Truth: wrist-mounted force sensor data

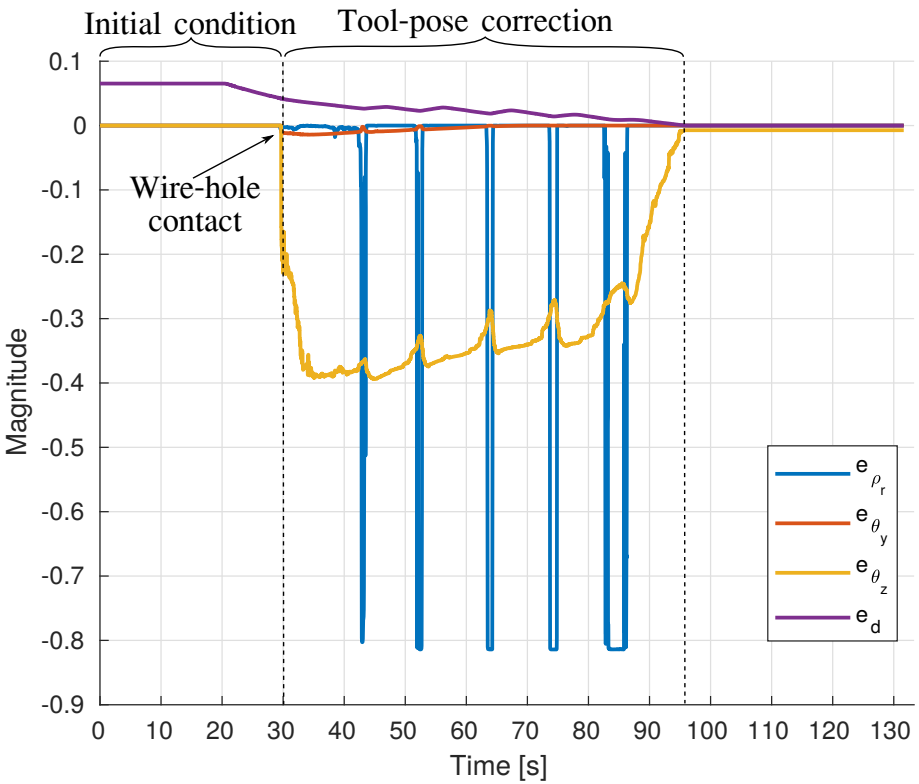
Tool Pose correction

PI control actions:

- **translational component**: aims to prevent the wire curling and facilitate the rotational correction
- **rotational component**: aims to minimize the force component normal to the hole axis.



Experimental Results

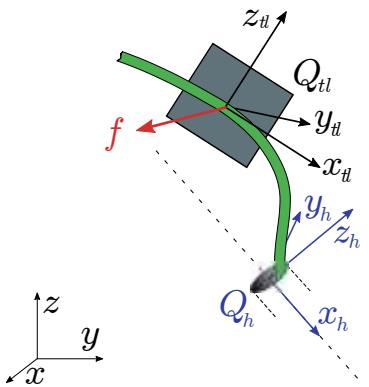


$$e_{\rho_r}(t) = \left| \min\left(0, \mathbf{x}_h^\top R_{tl}(t)\mathbf{f}(t)\right) \right|$$

$$e_{\theta_y}(t) = R_{tl}(t)\mathbf{z}_{tl}(t)^\top \mathbf{f}(t)$$

$$e_{\theta_z}(t) = R_{tl}(t)\mathbf{y}_{tl}(t)^\top \mathbf{f}(t)$$

$$e_d(t) = \|\mathbf{p}_h - \mathbf{p}_{tl}(t)\| - \Delta_p^*$$



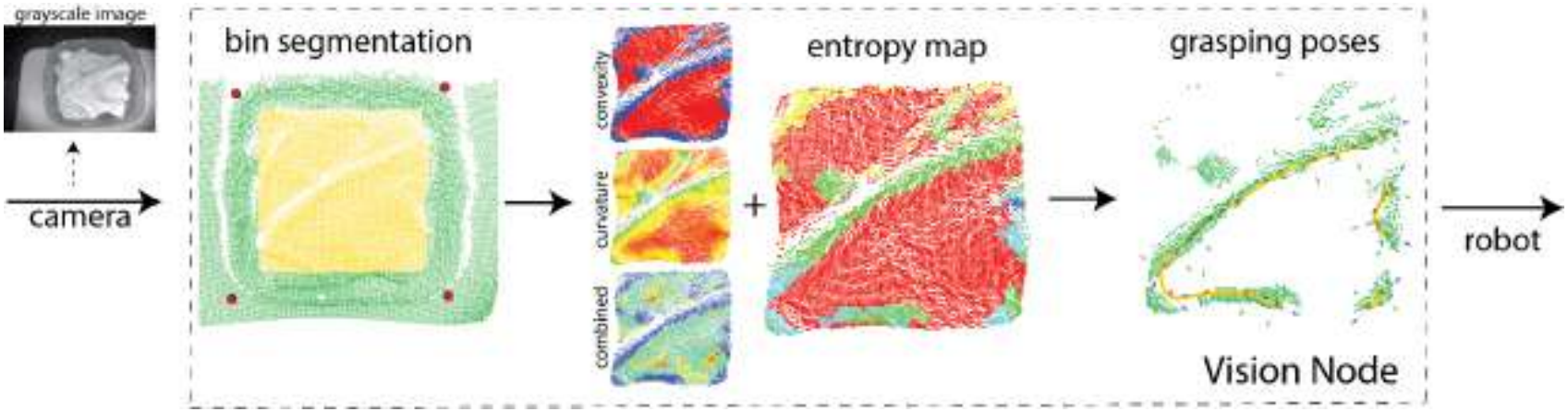
VIDEO

Extension to Deformable Planar Objects

Manipulation of Deformable Planar Objects

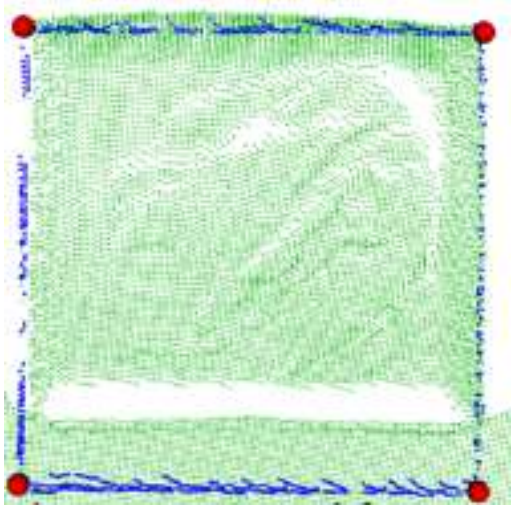
Pointcloud-based Identification of Optimal Grasping Poses for Cloth-Like Deformable Objects [Caporali and Palli, ETFA2020]

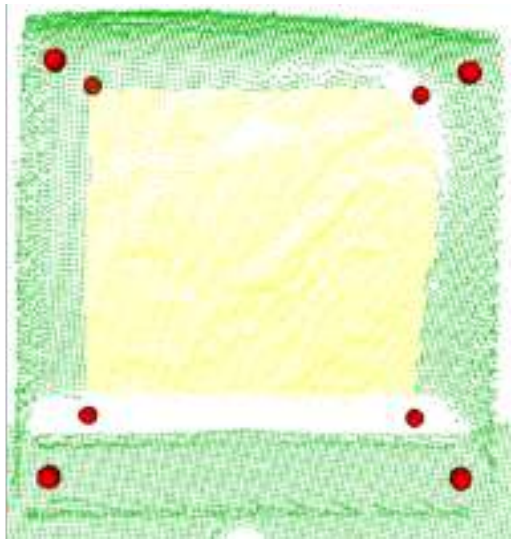
- pointcloud segmentation to extract cloth-related points
- detection of the graspable regions by exploiting an entropy measure combined with convexity and depth maps to increase the detection robustness
- fit of piecewise curves to identify the wrinkles
- estimates the optimal grasping pose for each detected wrinkle



Pointcloud Segmentation

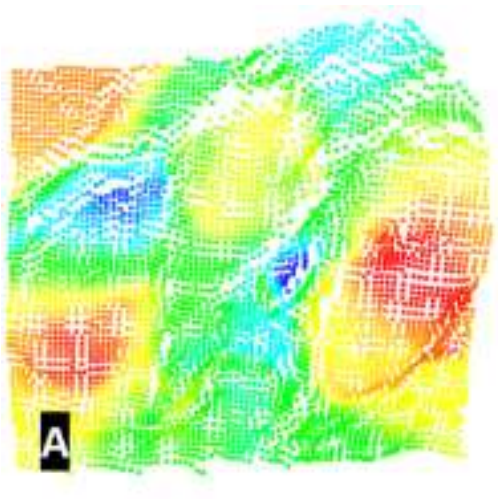
- Ambiguous points that do not belong to a cloth are removed
- The overall size of the pointcloud is reduced and, consequently, the computation efficiency increased
- Areas too close to the bin wall are discarded adding a safe space for the gripper operations (only for bin segmentation)



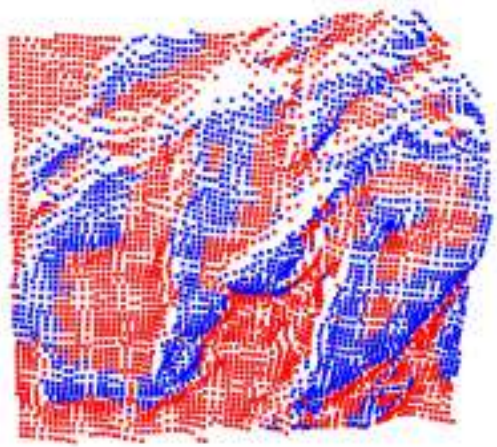


Graspable Regions Detection

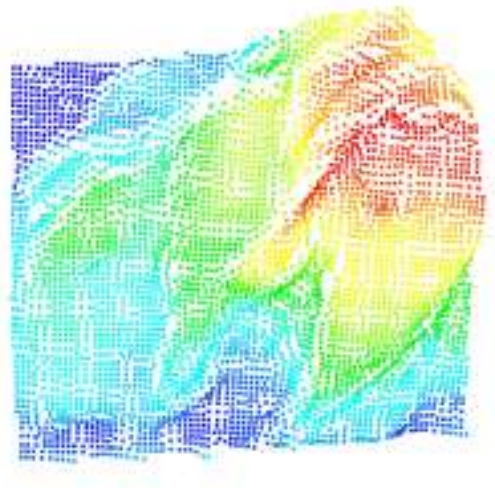
A graspability measure is employed to gain an understanding of the location of highly wrinkled areas in the cloth. This information is encoded into an *entropy map*. A *depth map* and *convexity map* are used as auxiliary cues to robustify the detection of the wrinkled areas



Entropy Map



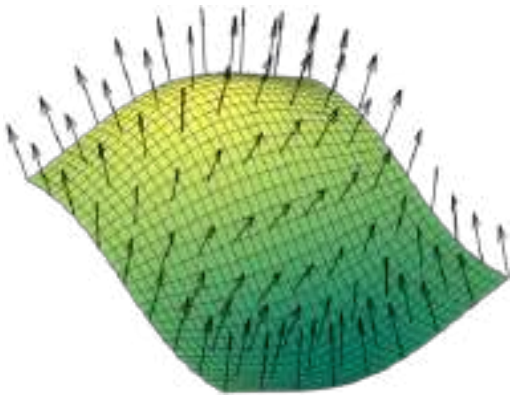
Convexity Map



Depth Map

Normals Estimation

Normals Estimation is computed using Moving Least Squares algorithm which smooth out the pointcloud surface by fitting a 2D spline curve



Normal vectors in Cartesian coordinates (n_x, n_y, n_z) are computed from the 2D spline curve transformed in spherical coordinates (ϕ, θ) (azimuth and inclination)

$$\phi = \text{atan}\left(\frac{n_z}{n_y}\right), \quad \theta = \text{atan}\left(\frac{\sqrt{n_z^2 + n_y^2}}{n_x}\right)$$

Entropy Filter

The entropy filter is used to discover regions of the clothes with a sparse distribution of normals

The entropy measure is defined as

$$H(x) = -w_x \sum_{i=1}^{n} p(x_i) \log p(x_i)$$

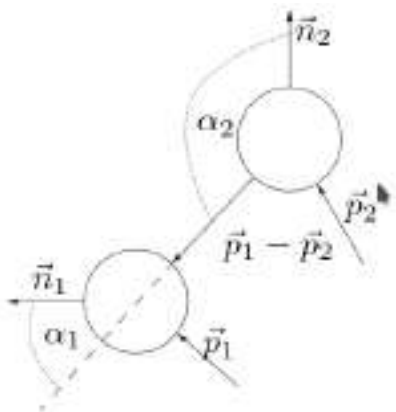
where x is the point considered to which a two-dimensional histogram of orientation angles in spherical coordinates (azimuth and inclination) is associated. The histogram is made of n bins for each dimension. The parameter w_x is the weight related to point x coming from the depth map. With x_i we are denoting the i-th bin of the histogram and with $p(x_i)$ its associated value. As the point is far away from the reference plane, its associated intensity value is larger and the weight factor increases. The input pointcloud is segmented based on an entropy threshold.

Convexity Map

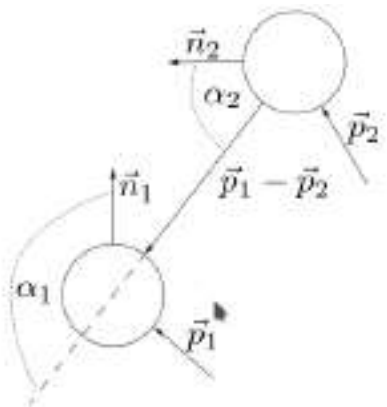
Concavity or convexity are important to define graps regions

Let's denote with $\vec{p}_1$ the (x, y, z) coordinates of the considered point and by $\vec{n}_1$ its normal, $\vec{p}_2$ and $\vec{n}_2$ be in turn the ones of its neighborhood points. The distance vector between $\vec{p}_1$ and $\vec{p}_2$ is computed as $\vec{d} = \vec{p}_1 - \vec{p}_2$
The angle α_1 between $\vec{n}_1$ and $\vec{d}$ is compared with the one α_2 between $\vec{n}_2$ and $\vec{d}$. The convexity condition can be expressed as

$$\alpha_1 < \alpha_2 \Rightarrow \cos(\alpha_1) - \cos(\alpha_2) > 0 \Rightarrow \vec{n}_1 \cdot \hat{d} - \vec{n}_2 \cdot \hat{d} > 0, \ \hat{d} = \frac{\vec{p}_1 - \vec{p}_2}{||\vec{p}_1 - \vec{p}_2||}$$



Convex



Concave

Depth Map

The depth map importance is twofold

- it is used for the calculation of the weight factor in case the weighted version of the entropy formula
- it allows to correctly detect situations in which the target cloth lies completely flat on a planar surface with not detectable wrinkles

The depth map is build by using a reference plane and by evaluating the distance between each point in the input pointcloud and this plane

The choice of the reference plane depends on the scenario. In the bin picking, the knowledge of the four top vertices location is used for the computation of the bin-top plane

Wrinkles Curve Fitting

The detection is performed on the segmented entropy map pointcloud of the previous step

P_{NN} denotes the neighborhood points of p_s (search point), p_C and M_C are referred to the centroid and covariance matrix respectively, M_C is normalized and its eigenvalues and eigenvector are used to estimate the direction of propagation. An increasing motion is enforced by checking the angle between the new direction and the previous one. The step motion is performed from p_C. p_s^{NN} indicates the closest neighborhood point of p_s

Algorithm 1: Curves Fitting Algorithm

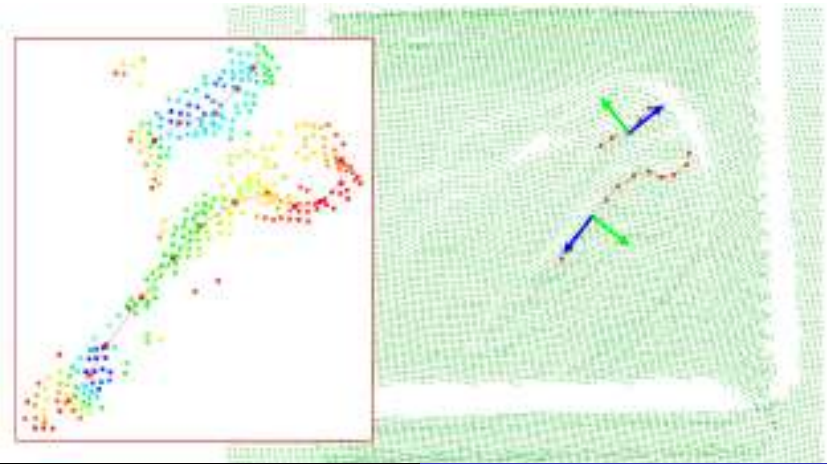
```
Result: Set of piecewise curves
initialization: p_s = global maximum entropy point;
for all possible high entropy areas do
    for it ≤ it_max do
        extraction of P_NN given p_s;
        calculation of p_C and M_C given P_NN;
        estimation of propagation direction;
        step in the estimated direction;
        update of p_s;
        if ||p_s^NN − p_s|| ≤ threshold then
            it++;
        else
            break;
        end
    end
    curve = concatenation of p_C points;
    extraction of curve area from entropy map;
    p_s = new global maximum entropy point;
end
```

Grasp Poses Estimation

Each extracted wrinkle is stored as a set of sequence of points B that are then connected to obtain a piecewise representation of a curve denoting the wrinkle's path

The optimal graping is attached to the curve saction having lower curvature with the first axis is aligned to the wrinkle direction while the second one is set orthogonal to the first one. The third axis is orthogonal to both the first and second pointing downward

The grasp poses can be further ordered according to their distance from the bin border in order to select the safer for the robot to execute the grasp

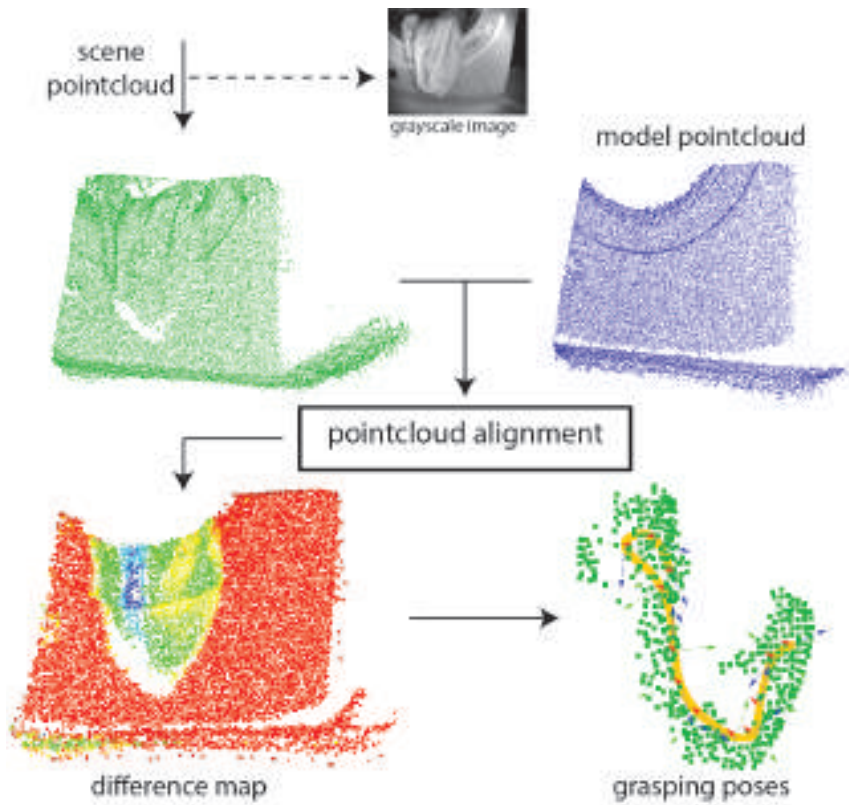


Extension to Washing Machines

The method is also applied to recovery graps from the washing machine borders [Caporali, Bedada and Palli, ICPS2021]

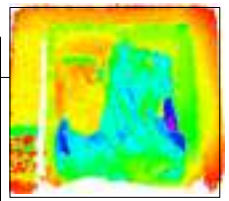
The procedure is composed by:

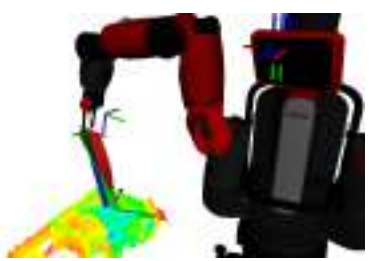
- **Pointclouds Registration**, to align the scene with the washing machine model
- **Difference Map**, to remove the scene pointcloud and detect the presence of hanging laundry
- **Grasping Poses Generation**, consists in applying the graps detection algorithm to the difference map



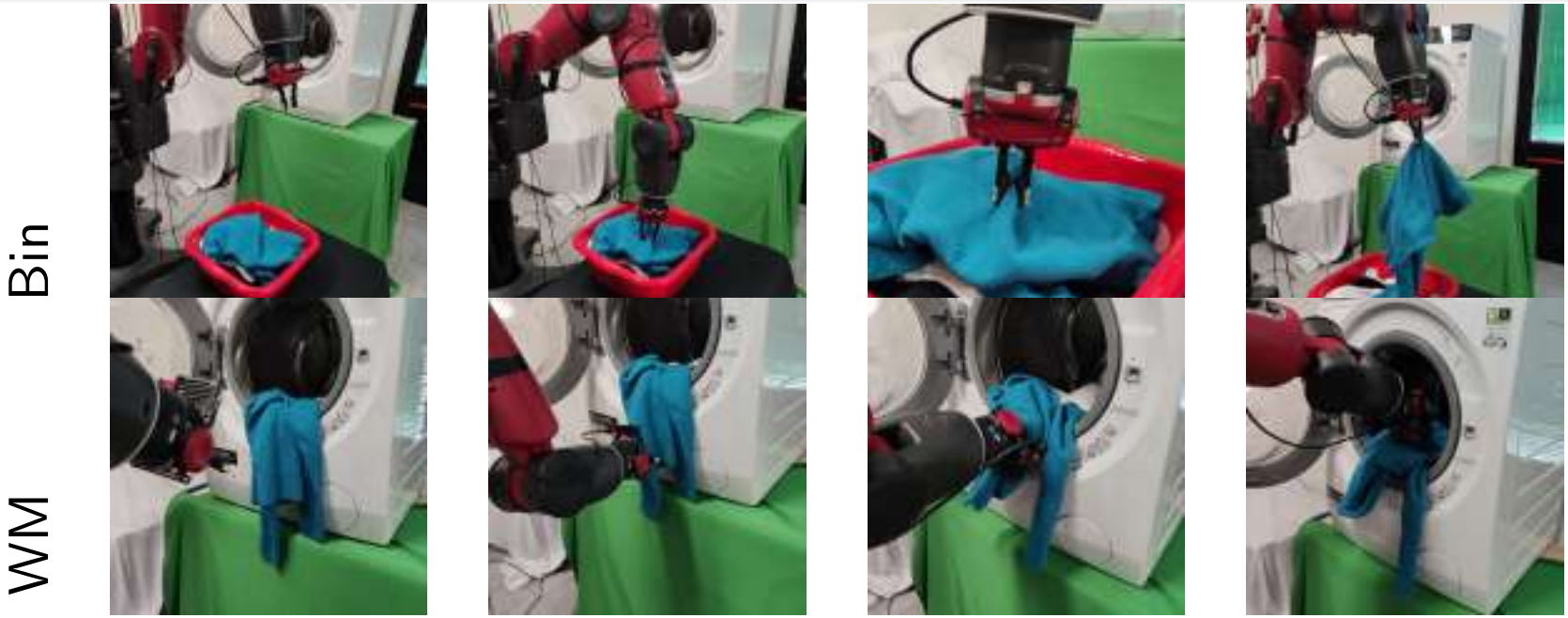
Experimental System

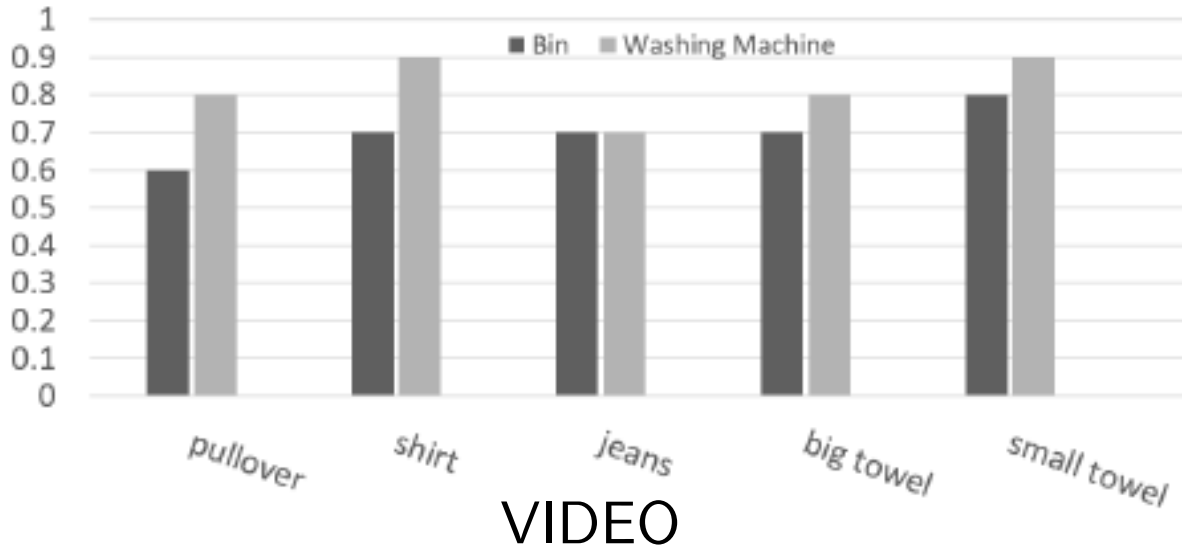

Pico_flexx_camera


Pointcloud



Experimental Results





Thanks!