Human initiated grasp space exploration algorithm for an underactuated robot gripper using variational autoencoder

Clement Rolinat, Mathieu Grossard, Saifeddine Aloui, Christelle Godin

Human Initiated Grasp Space Exploration Algorithm for an Underactuated Robot Gripper Using Variational Autoencoder

Clément Rolinat, Mathieu Grossard
Université Paris-Saclay, CEA, List,
F-91120, Palaiseau, France
{clement.rolinat, mathieu.grossard}@cea.fr

Saifeddine Aloui, Christelle Godin
Université Grenoble Alpes, CEA, Leti,
F-38000, Grenoble, France
{saifeddine.aloui, christelle.godin}@cea.fr

Abstract— **Grasp planning and most specifically the grasp space exploration is still an open issue in robotics. This article presents an efficient procedure for exploring the grasp space of a multifingered adaptive gripper for generating reliable grasps given a known object pose. This procedure relies on a limited dataset of manually specified expert grasps, and use a mixed analytic and data-driven approach based on the use of a grasp quality metric and variational autoencoders. The performances of this method are assessed by generating grasps in simulation for three different objects. On this grasp planning task, this method reaches a grasp success rate of 99.91% on 7000 trials.**

Index Terms— **multifingered gripper, grasp space exploration, variational autoencoder, grasp quality metric**

I. INTRODUCTION

Grasping is fundamental in most of the industrial manufacturing processes such as pick-and-place, assembly or bin picking tasks. The grasp planning question is still an active research topic. It aims at finding a gripper configuration that allows to grasp an object reliably. This grasp configuration needs to be kinematically reachable and collision free with respect to the environment, and the produced grasp needs to be stable and robust to external perturbation. Finding such a grasp configuration requires to explore the grasp space, that is the subset of gripper configurations that effectively grasp the object. Thus, grasp planning is both object-dependent and hardware-dependent. Taking into account those constraints during the exploration is not trivial, as objects can have complex shapes, and gripper-arm combination can have complex kinematics.

This is even more true for underactuated or compliant architectures, which are often chosen for grasping tasks [1]. Indeed, such architecture allows to reduce the controller complexity by reducing the number of controlled degrees of freedom, while retaining sufficient kinematic abilities. Moreover, it tends toward producing robust grasps by their mechanical structure.

The grasp planner should be able to find in the high dimensional and highly constrained grasp space a configuration that fulfills a given criterion. There are two main ways to achieve this: analytic approaches and data-driven approaches [2]. Analytic approaches rely on an analytic description of the grasping problem [3][4][5]. Data-driven approaches depend on machine learning methods to predict grasps from object depth map or point cloud [6][7][8][9][10][11][12].

A shared issue is the grasp dataset creation, that is the grasp space exploration. A variety of high quality grasps need to be discovered by exploring the space of possible grasp configurations. There is two main approaches regarding this exploration [5]: contact point approaches, and gripper configuration approaches. In the first case the grasp space exploration comes down to test various combination of contact point location on the object surface. However there is no guarantee that a given combination is kinematically admissible for a given gripper, and the inverse kinematics can even be intractable for underactuated or adaptive grippers. In the second case, the grasp space is explored by testing several gripper spatial configurations. This is more suited for underactuated gripper. Nevertheless, there is no assurance that a given gripper configuration is a priori able to grasp the object without realizing extensive simulation trials beforehand [11][12].

To circumvent this dimensionality issue related to the huge size of the grasp space, numerous contact point approaches limit their search to fingertip contacts [4][6], and gripper configuration approaches often use a bi-digital gripper and limit their search to planar grasps [7][8][9][10]. For more complex grippers, as multifingered and adaptive ones, a human input is often required. For example, in Santina et al. [13], authors identified a set of ten grasp primitives from human examples, and reduced the grasp space to those primitives only. In Choi et al. [14], authors choose to limit the search space by discretizing it.

The contribution of this article is a procedure that allows to explore efficiently the grasp space of a multifingered and underactuated gripper. It relies on a limited set of object-dependent primitive gripper configurations, that are likely to grasp the object based on human experience, around which the exploration is focused. This allows to reduce the search space dimensionality, without restricting the kinematic potential of the gripper with arbitrary and strict hypothesis such as fingertip contacts or planar grasps. In this article, this procedure is applied in simulation on three different objects, and allows to successfully generate relevant grasps.

Section II is dedicated to the problem statement and a presentation of the used framework. Then, in section III the general workflow will be explained. In section IV some implementation details will be given. Finally, the resulting grasp space exploration algorithm will be tested on grasp planning trials in section V. To conclude, this work will be discussed and the planned future works will be presented.

II. PROBLEM STATEMENT & TOOL USED

A. Simulation Setup

The gripper simulated in this work has three fingers, and is underactuated and adaptive. The compliance and underactuation allow it to naturally adapt itself to the object geometry, without the need to carefully control each joint, thus increasing the robustness of the grasp.

This gripper has two joints on each finger and one actuator per finger to control both joints. The second (distal) phalanx starts moving when the applied effort on the finger is above a given force threshold. A fourth actuator allows a coupled and symmetrical abduction-adduction (spread) motion of two fingers.

This gripper is mounted as end effector of a six degrees of freedom industrial robot arm.

The simulation setup described above is implemented with Gazebo simulator [15]. A picture of this simulated setup is displayed in Figure 1.

B. Problem Statement

An object is placed on a table in the workspace of the considered robotic setup. It is assumed that the object is known, as well as the pose in the scene of its associated frame F_{obj}. Knowing the pose is not a strong hypothesis, as there exists methods to extract pose information of known objects from a point cloud, for example [16].

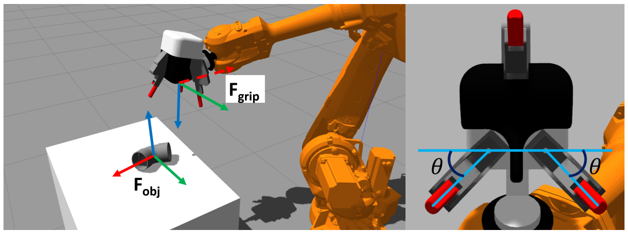
The goal of the grasp space exploration is to find grasp configurations with a high quality. The metric used to assess grasp quality will be described in subsection II-C. In this work, a grasp configuration is a gripper configuration that is able to grasp the object without colliding with the table.

A gripper configuration is defined as follows by eight parameters:

- the pose of the gripper frame F_{grip},
  $$(x, y, z, q_x, q_y, q_z, q_w) \in \mathbb{R}^3 \times SO(3) = SE(3)$$
  with the orientation expressed in quaternion convention;
- the abduction-adduction motion, or spread angle θ, as shown on Figure 1.

The dimensionality of this configuration space is high, but this allows to fully leverage the grasping ability and kinematic potential of the gripper. Thus, the grasp space is a subset of this configuration space, with an additional constraint that every gripper configuration is able to grasp the object without colliding with the table.

To locate the gripper, a dedicated frame F_{grip} situated between the fingers in front of the palm is used. This frame is displayed on Figure 1. Gripper poses are expressed relatively to the object frame F_{obj}, in order to be invariant to object poses.



(a) Pose of the frame F_{grip} used to locate the gripper relatively to the object frame F_{obj}.

(b) Spread angle θ.

Fig. 1. Gripper frame and spread angle.

C. Analytical Grasp Quality Metric

A grasp has several properties which can define its quality [17]. One of them is the *force-closure* property, that is the ability to resist external disturbances in any direction. See Murray et al. [18] and Prattichizzo et al. [19] for more in depth mathematical description. In particular, it relies on the concept of grasp map G, a matrix that stores geometrical information about the grasp.

Several metrics have been developed from the computation of the grasp map. Some of them are considering algebraic properties of the matrix, such as the full rank of the G matrix, and can be used as a proxy for force-closure. In this paper, the *minimum singular value of* G, Q_{MSV}, has been chosen. It is worth noting that the proposed grasp space exploration method could work with an other grasp quality metric. With $\sigma(G)$ the vector of singular values of G, Q_{MSV} can be simply expressed as follows:

$$Q_{MSV} = min\left(\sigma\left(G\right)\right)$$

Q_{MSV} greater than 0 is a necessary condition for force-closure. The greater Q_{MSV} is, the farther the grasp is from a numerical singularity. However, this is not a sufficient condition. In the general case, it is difficult to assess if a grasp is force-closure because it comes down to an optimization problem with non-linear constraints.

D. Variational Autoencoders

A variational autoencoder (VAE) allows to generate consistent data from its latent space more reliably than a classic autoencoder, which is designed to learn a compressed representation of data.

In a VAE, among other features, a supplementary term is added in the loss function: the Kullback-Leibler (KL) divergence [20]. This term helps the data to be represented as a normal distribution in its latent space, and thus regularizing it.

III. GENERAL WORKFLOW

The idea presented in this paper is to take advantage of the human ability to find promising grasp configurations. Indeed, it is easy for a human to find gripper configurations that are likely to grasp a given object, that is configurations belonging to the grasp space. However, those primitive grasps do not necessarily have a high quality. Indeed, it is difficult for a human to assess a priori the relative and absolute

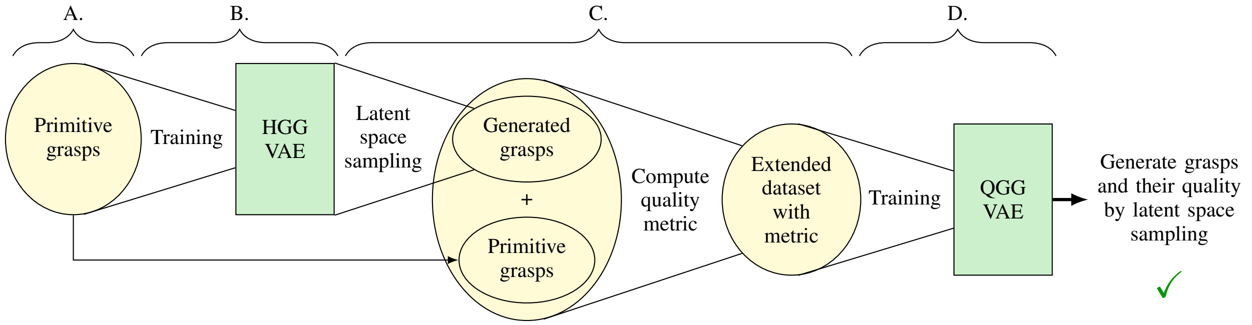


Fig. 2. Scheme of the presented workflow.

quality of grasp configurations. Here, it is the role of the space exploration to focus around those primitive grasps to constitute a collection of grasps with various quality, and discover grasps with higher quality than the primitive ones if such grasps exist.

The general workflow used to achieve this is described below, and is decomposed as follows:

A. the constitution of a primitive grasp dataset
B. the training of a Human-initiated Grasp Generator Variational Autoencoder (HGG)
C. a dataset extension & grasp quality estimation phase
D. the training of a Quality-oriented Grasp Generator Variational Autoencoder (QGG)

This procedure is summarized on Figure 2.

A. Primitive Grasp Dataset

To leverage the human ability to find gripper configurations belonging to the grasp space, an object-dependent primitive grasp dataset is built. Concretely, a primitive grasp is a handcrafted gripper configuration, with its pose and spread angle human-chosen so that it is collision free and likely to grasp the object. The spread angle (shown in Figure 1) is chosen between four discrete values corresponding to main gripper internal layouts: $\theta = 0$, $\theta = \pi/6$, $\theta = \pi/4$, and $\theta = \pi/2$.

The dataset stores the eight parameters describing each primitive grasp along with the four parameters of the tabletop plane Cartesian equation in object frame. Indeed, many objects have different possible stable positions on the table. This is a critical information to avoid collisions with it. Some grasps may collide with the table in a given stable position, while being suitable for an other stable position. Expressing the grasp configuration in the object frame is still useful as it allows an invariance to a position change and to a rotation around a vertical axis.

B. Human-initiated Grasp Generator Variational Autoencoder (HGG)

The goal of the Human-initiated Grasp Generator VAE (HGG) is to extract the correlations existing between the parameters of different grasp primitives. Such correlations exist because primitive grasps are in the grasp space, and the grasp space is a subset of the gripper configuration space. A VAE is able to use those correlations to build an efficient mapping of the grasp space in its latent space, with fewer parameters than the initial gripper configuration space. This efficient dimension reduction allows to generate grasps sufficiently close to primitive grasps to remain pertinent, while exploring the configuration space around it, by simply sampling in the VAE latent space. In this case, a mono-dimensional latent space has been chosen, because it allows the strongest compression. The main drawback is that if the true dimensionality of the grasp space is greater than one, there will be information loss during compression. The effect of a latent space of higher dimension on the grasp space exploration needs to be investigated in future work.

The HGG is trained on the primitive grasp dataset. Its inputs and outputs are summarized on Figure 3.

inputs											
gripper configuration in frame F_{obj}								tabletop plane Cartesian equation in frame F_{obj}			
x	y	z	q_x	q_y	q_z	q_w	θ	a	b	c	d
outputs											

Fig. 3. HGG inputs and outputs data. This input-output architecture is similar to Conditional VAE architecture introduced in [21]. The generated grasp configuration is conditioned by the tabletop plane equation.

For the gradient descent during the training, a Mean Square Error (MSE) is computed for each gripper parameter.

Each of these errors is averaged on each batch. Then, the global loss for each batch is computed as the sum of these averaged errors together with the KL-divergence loss.

C. Dataset Extension & Grasp Quality Estimation

Sampling in the latent space of the HGG allows to explore the grasp space. This sampling is more efficient than a sampling around expert grasp in the gripper configuration space. Indeed, the HGG takes into account the correlations existing between the parameters of the grasp configurations, and thus maps the grasp space.

Each sampled configuration is then tested in simulation along with each primitive grasp to check its success.

A configuration is successful if the following conditions are met:

- it does not collide with the table
- it successfully lifts the object from the table
- its Q_{MSV} is greater than 0.

For each successful configuration, the computed Q_{MSV} quality value is registered. For failed configurations, a null value is registered as quality value.

This allows to extend the primitive dataset by exploring extensively the grasp space. Thus, a collection of grasps with various quality values can be constituted, and if better grasps than the primitive ones exist, they can be discovered.

D. Quality-oriented Grasp Generator Variational Autoencoder (QGG)

The goal of the Quality-oriented Grasp Generator VAE (QGG) is to reliably generate grasps with their corresponding grasp quality.

The QGG is trained on the extended set formed by merging the primitive grasp set with the generated grasp set (both successful and failed). Learning failed grasps together with successful ones reduces the risk of predicting a high quality for a failing grasp when some failing configurations are close to successful ones. The inputs-outputs are the same as for the HGG with the grasp quality added as a supplementary output. This way, the QGG decoder learns to predict the grasp quality while reconstructing the other grasp configuration parameters. Moreover, the dataset extension allows to represent more accurately and more reliably the grasp space.

The latent space dimension and the loss function are the same as the ones used for the HGG. Regarding the grasp quality, a MSE is computed and added to the loss.

The QGG can be used to generate high quality grasps by sampling in its latent space and selecting only grasps having their quality above a given threshold.

IV. WORKFLOW IMPLEMENTATION

The workflow described above is performed on three different objects. Implementation details will be given in this section.

A. Objects & Primitive Grasps

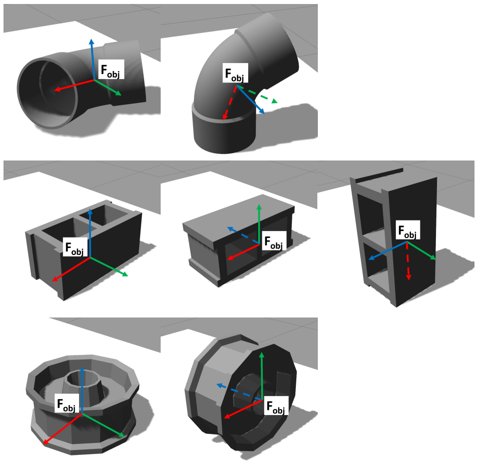


Fig. 4. The chosen objects and their frame F_{obj} in their different stable positions: bent pipe (first row), cinder block (second row), and pulley (third row)

The chosen objects are:

- a connector bent pipe
- a pulley
- a small cinder block

Their CAD model used in the simulation in their different stable positions are visible on Figure 4. Those object were chosen for their relative complexity and diversity in term of shapes.

A set of primitive gripper configurations is determined for each of those objects for each of their stable position. These primitive gripper configurations can be sorted in different grasp types presented on Figure 5. For each of these grasp type, several variants are created.

For each object is gathered the following number of primitive grasps:

- bent pipe: 145 samples
- cinder block: 141 samples
- pulley: 118 samples

Around one hour is needed for a given object to register all those primitives.

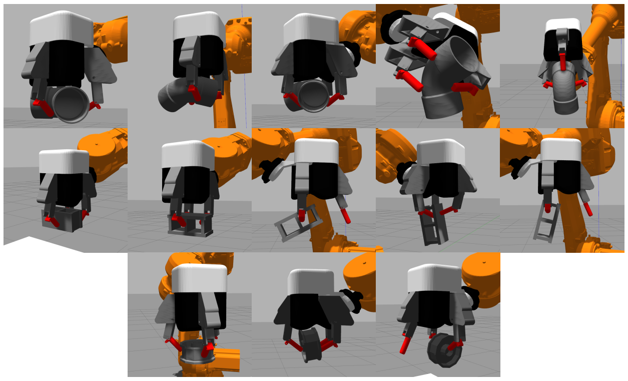


Fig. 5. Primitive grasp types for the three chosen objects. On the first row the grasp types for the bent pipe, on the second row for the cinder block, and on the third row for the pulley

B. VAEs Training & Quality Metric Computation

The architecture used for both HGG and QGG is displayed in Figure 6.

before the training, the inputs and outputs data are normalized. This allows a faster training as the network does not have to scale its data by itself.

To make sure that the quaternion outputs by the decoder is a unit quaternion, a custom activation function is used to normalize the quaternion on the output layer of the decoder.

For each object, a HGG network is trained on the primitive grasp dataset, with the input-output architecture described in Figure 3.

After the HGG training, 2000 grasps for each stable position of each object are generated by sampling in the HGG latent space and tested in simulation to compute their quality metric, in order to create the extended dataset for the QGG training. Some statistics about generated and primitive grasps quality are summarized on Figure 7.

The grasp quality mean and median of the primitive and generated grasps are close. This is expected as the VAE tries

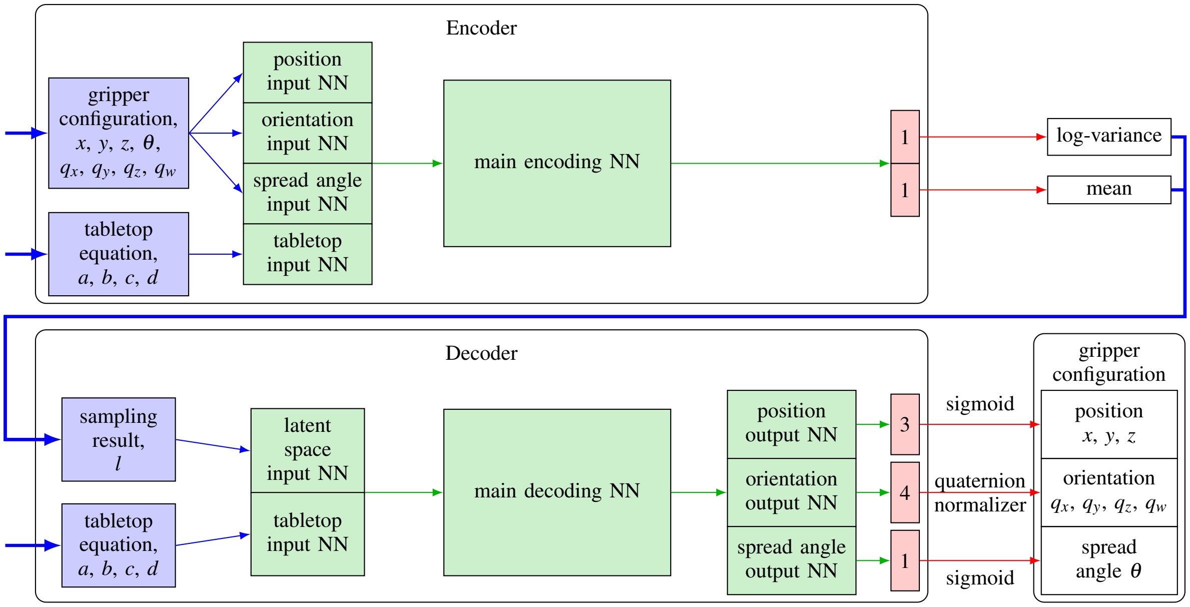


Fig. 6. HGG architecture. In blue the input layers, in green the hidden neural networks (NN) and in red the output layers. The hidden NN inner layers are fully connected layers, with hyperbolic tangent activation functions. The main encoding and main decoding NN have symmetrical inner architecture. The latent space dimensionality is one. The supplementary input for the tabletop equation allows to ensure that the generated grasp depends on it [21]. This architecture is implemented with Tensorflow [22] and Keras [23] python libraries. The QGG has the same architecture, with an added output to the decoder for the grasp quality, with its associated output layer and hidden output NN. HGG and QGG have around 12000 trainable parameters each.

to reproduce the underlying distribution of the learning set. The goal is to explore the grasp space around the primitive grasps, and discover a collection of grasps with various quality.

			generated set	**primitive set**
bent pipe	total number of grasps		4000	145
	number of successful grasps		2845	141
	metric statistics	median	0.1022	0.1018
		mean	0.1035	0.1047
		maximum	0.2128	0.2257
cinder block	total number of grasps		6000	141
	number of successful grasps		5690	141
	metric statistics	median	0.0675	0.0670
		mean	0.0535	0.0563
		maximum	0.1877	0.1041
pulley	total number of grasps		4000	118
	number of successful grasps		3591	111
	metric statistics	median	0.0739	0.0730
		mean	0.0653	0.0648
		maximum	0.3115	0.1195

Fig. 7. information summary about primitive and HGG generated grasps (step C of the workflow). The metric statistics are taking into account successful grasps only.

For Two objects, a global maximum better than the primitive grasps is found in the generated grasps. Indeed, the VAE learns to interpolate between the primitive grasps: in case a better grasp is "between" two primitive grasps, it is able to generate it.

Regarding the bent pipe, no generated grasps are better than the best primitive grasp. Two possible explanations are:

- The global maximum may already be in the primitive dataset. It is not unlikely, as it is a human-crafted set of configuration, and humans tend to produce high quality grasps.
- Some configurations of the dataset, among which primitives with best quality, are reconstructed poorly because of the trade-off between reconstruction and KL-divergence. The model may not generate configurations sufficiently close to those primitives to efficiently explore this part of the grasp space. This may be due to a too high compression, linked to a too small latent space.

Regarding the QGG network, it is trained for each object on the extended dataset constituted of the primitive grasps and the HGG generated grasps. Its performances are assessed on a grasp planning task in section V.

C. QGG Latent Space

The Figure 8 shows how the QGG network has extracted the correlations between the gripper configuration parameters and the grasp quality value for the pulley object. The primitive grasps are evenly spread in the latent space, which is a direct consequence of the KL-divergence loss component. This is at the cost of a higher reconstruction error, but allows to sample in the latent space safely, without producing inconsistent configuration.

Due to contact points volatility in simulation, the computed metric has a variability and is not fully deterministic for a given configuration. Despite that, the QGG has successfully captured the overall tendency, without over-fitting on the noise.

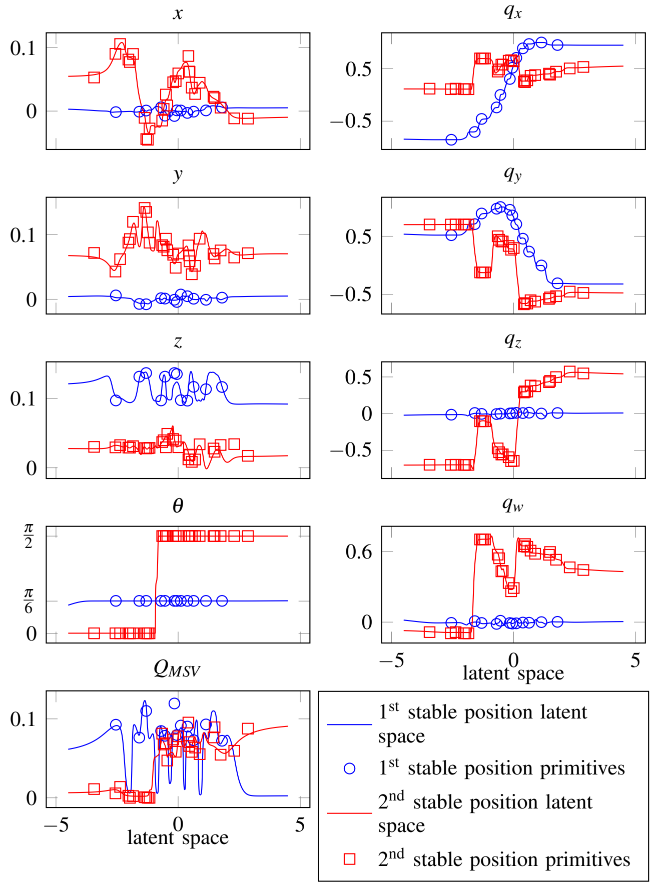


Fig. 8. QGG latent space representation for the pulley object. The curves are obtained by sampling values in $[-4.5, 4.5]$ and passing them through the decoder. The outputs are gripper configurations stored in the latent space of the QGG. The scatter plots are obtained by passing some pulley primitive configurations through the encoder (only one third of the primitives displayed for readability).

V. GRASP PLANNING TRIALS

To validate the grasp space exploration workflow, grasp planning trials are conducted on each object. The grasp planning algorithm is described in Algorithm 1.

This planning procedure is executed on 1000 distinct object poses for each stable position of each object. The position of the object frame projection on the tabletop plane is chosen randomly inside a 10×10 centimeters square, and its orientation relative to the vertical axis is also drawn randomly between 0 and 2π.

Algorithm 1 Grasp planning algorithm.

```
 1: grasp candidate list ← ∅
 2: while length(grasp candidate list) < 3 do
 3:     configuration ← QGG decoding of a sampled value in its latent space
 4:     if configuration predicted grasp quality > threshold then
 5:         if configuration is collision free and kinematically reachable then
 6:             append configuration to grasp candidate list
 7:         end if
 8:     end if
 9: end while
10: execute the grasp with highest predicted grasp quality among grasp candidate list
```

Three parameters are monitored to assess the performances of the presented workflow:

1) the grasp success rate.
2) the number of collision and reachability checking iterations needed to find three admissible grasps (Algorithm 1 line 5), as it is the most time consuming step. Indeed, the presented workflow is object-centric. It does not take into account the arm kinematic and environment, so depending on the object pose in the robot workspace, the probability of sampling an admissible configurations in the QGG latent space vary.
3) the grasp quality relative prediction error.

These parameters are shown on Figure 9.

	1) success rate (%)	**2) Algorithm 1 line 5 mean iterations**	**3) mean quality prediction error (%)**
bent pipe	100	7.6	15.6
cinder block	100	5.8	7.5
pulley	99.7	7.4	14.8

Fig. 9. Performances on grasp planning trials.

The mean relative prediction error has the same order of magnitude than the computed metric noise.

The low number of collision and reachability checking iterations shows that all grasp types and their variants are evenly distributed in the latent space. Indeed, some grasp types or variants within a grasp type are reachable only for some object poses relative to the robot.

The low failure rate shows that the procedure presented in this work successfully explore and reproduce the grasp space, as it is able to generate reliably high quality grasps for various object poses.

VI. CONCLUSION

This work presents an efficient method for grasp space exploration. It explores a high dimensional grasp space by focusing the search around human inputs, and take into account analytic grasp quality criterion. This procedure was then used to successfully plan grasps in simulation.

Various tracks can be investigated in future works. First, the effect of a latent space of higher dimension on the grasp space exploration needs to be assessed. Indeed, using a larger latent space may improve the exhaustiveness of the exploration by reducing information loss due to compression. Moreover, a reduction of the number of human inputs required per object would be useful to scale this method to several objects. Furthermore, in this study, the grasp planning performances were assessed in simulation only. These results need to be confirmed on a real setup. Finally, this space exploration procedure could be used to constitute a grasp dataset with high quality grasps to be learned by a data-driven grasp planner. Indeed, this may allow to generalize to unseen objects.

References

[1] W. Townsend, “The BarrettHand grasper – programmably flexible part handling and assembly,” *Industrial Robot*, vol. 27, no. 3, pp. 181–188, 2000.

[2] A. Sahbani, S. El-Khoury, and P. Bidaud, “An overview of 3d object grasp synthesis algorithms,” *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 326–336, 2012.

[3] D. Berenson, R. Diankov, Koichi Nishiwaki, Satoshi Kagami, and J. Kuffner, “Grasp planning in complex scenes,” in *IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 42–48.

[4] M. A. Roa, R. Suarez, and J. Rosell, “Grasp space generation using sampling and computation of independent regions,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008, pp. 2258–2263.

[5] Z. Xue, J. M. Zoellner, and R. Dillmann, “Grasp planning: Find the contact points,” in *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2007, pp. 835–840.

[6] Z. Zhao, W. Shang, H. He, and Z. Li, “Grasp prediction and evaluation of multi-fingered dexterous hands using deep learning,” *Robotics and Autonomous Systems*, vol. 129, p. 103550, 2020.

[7] L. Pinto and A. Gupta, “Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 3406–3413.

[8] A. Depierre, E. Dellandréa, and L. Chen, “Jacquard: A large scale dataset for robotic grasp detection,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 3511–3516.

[9] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, “Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection,” *The International Journal of Robotics Research*, vol. 37, no. 4, pp. 421–436, 2018.

[10] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J. A. Ojea, and K. Goldberg, “Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics,” in *Robotics: Science and Systems (RSS)*, 2017.

[11] M. A. c. Riedlinger, M. Voelk, K. Kleeberger, M. U. Khalid, and R. Bormann, “Model-free grasp learning framework based on physical simulation,” in *ISR 2020; 52th International Symposium on Robotics*, 2020, pp. 1–8.

[12] A. Mousavian, C. Eppner, and D. Fox, “6-dof graspnet: Variational grasp generation for object manipulation,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 2901–2910.

[13] C. D. Santina, V. Arapi, G. Averta, F. Damiani, G. Fiore, A. Settimi, M. G. Catalano, D. Bacciu, A. Bicchi, and M. Bianchi, “Learning from humans how to grasp: A data-driven architecture for autonomous grasping with anthropomorphic soft hands,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1533–1540, 2019.

[14] C. Choi, W. Schwarting, J. DelPreto, and D. Rus, “Learning object grasping for soft robot hands,” *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2370–2377, 2018.

[15] N. Koenig and A. Howard, “Design and use paradigms for gazebo, an open-source multi-robot simulator,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, 2004, pp. 2149–2154.

[16] B. Drost, M. Ulrich, N. Navab, and S. Ilic, “Model globally, match locally: Efficient and robust 3d object recognition,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 998–1005.

[17] M. A. Roa and R. Suárez, “Grasp quality measures: review and performance,” *Autonomous Robots*, vol. 38, no. 1, pp. 65–88, 2015.

[18] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[19] D. Prattichizzo and J. C. Trinkle, “Grasping,” in *Handbook of Robotics*. Springer, 2008, pp. 671–700.

[20] D. Kingma and M. Welling, “Auto-encoding variational bayes,” in *International Conference on Learning Representations (ICLR)*, 2014.

[21] K. Sohn, X. Yan, and H. Lee, “Learning structured output representation using deep conditional generative models,” in *International Conference on Neural Information Processing Systems*, ser. NIPS’15, vol. 2. Cambridge, MA, USA: MIT Press, 2015, pp. 3483–3491.

[22] M. Abadi *et al.*, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015, software available from www.tensorflow.org.

[23] F. Chollet *et al.*, “Keras,” 2015, software available from www.keras.io.