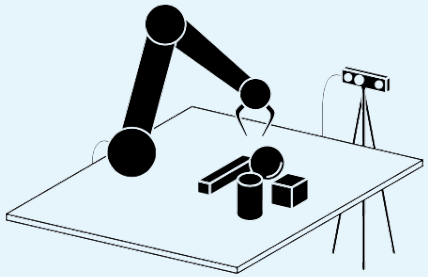


Autonomous Robotic Grasping

Final Year Project

Sri Aditya Deevi
SC18B080
ECE (Avionics)

Research advisor: Dr. Deepak Mishra

Indian Institute of Space Science and Technology

20 May, 2022

Outline of the Presentation

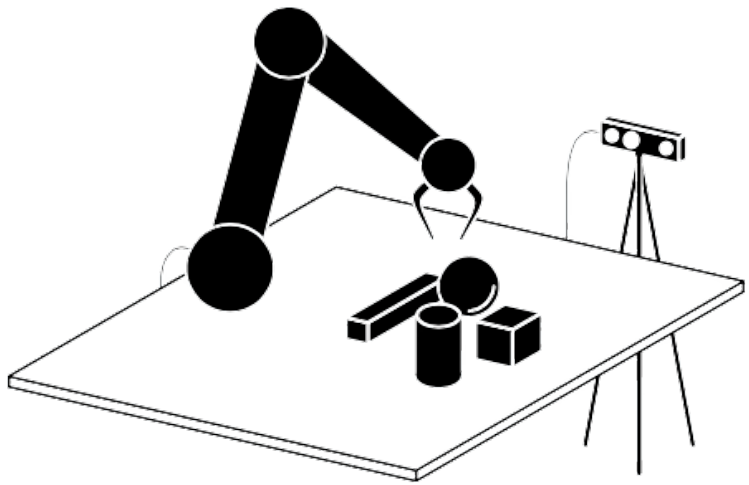
Problem Statement

Problem Statement

Two ARG tasks are considered, critical for developing "intelligent" robotics :

Task I : Grasping Various Objects in Diverse Environments

Task II : Dynamic Grasping of Moving Objects

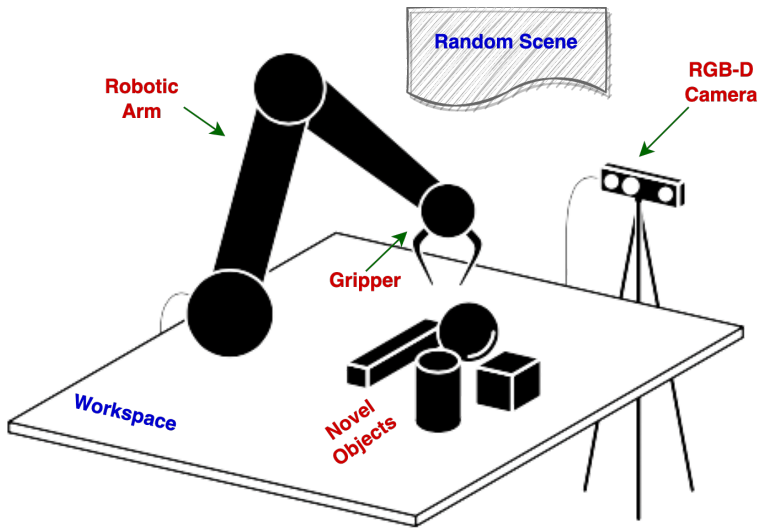


Problem Statement

Two ARG tasks are considered, critical for developing "intelligent" robotics :

Task I

Aims to use Deep Reinforcement Learning techniques to train a robotic arm to grasp novel objects, i.e. objects whose 3D model is not known apriori, in novel random scenes.



Task II : Dynamic Grasping of Moving Objects

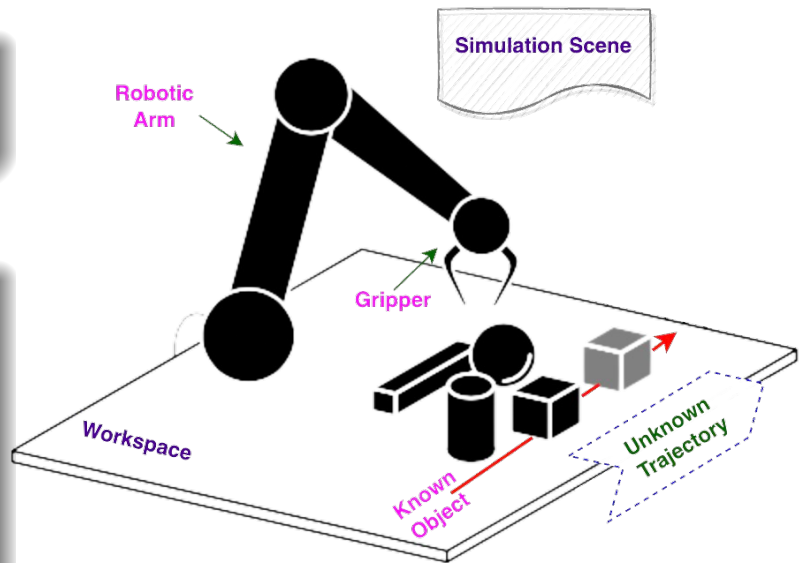
Problem Statement

Two ARG tasks are considered, critical for developing "intelligent" robotics :

Task I : Grasping Various Objects in Diverse Environments

Task II

Aims to use Deep Learning and Inverse Kinematics based Motion Planning techniques to train a robotic arm to grasp dynamic objects of interest, whose 3D model is known apriori but motion trajectory is not known.



Task I : Grasping Various Objects in Diverse Environments

Task Formulation

Formulated as an Episodic MDP, where the agent is a high level controller taking actions in cartesian space :

- *Episode Start* $\rightarrow$ New set of objects (3D Model is not known) given.
- *During Episode* $\rightarrow$ Aim is to grasp an object and lift in 12.5 cm above the ground.
- *Episode End* $\rightarrow$ Success (or) 100 time-steps {Failure}

Task Formulation

Formulated as an Episodic MDP, where the agent is a high level controller taking actions in cartesian space :

- *Episode Start* → New set of objects (3D Model is not known) given.
- *During Episode* → Aim is to grasp an object and lift in 12.5 cm above the ground.
- *Episode End* → Success (or) 100 time-steps {Failure}

Task Formulation

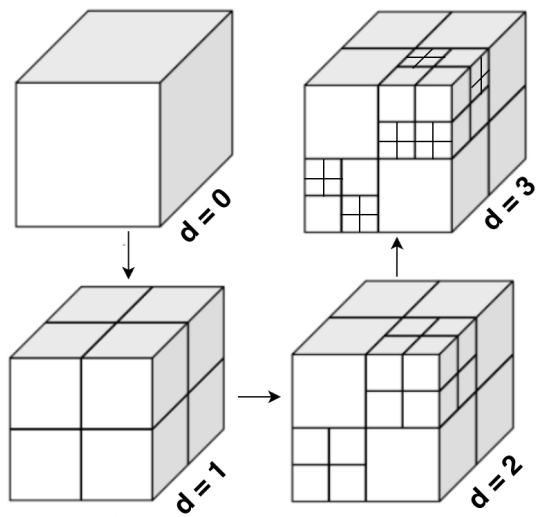
Formulated as an Episodic MDP, where the agent is a high level controller taking actions in cartesian space :

- *Episode Start* → New set of objects (3D Model is not known) given.
- *During Episode* → Aim is to grasp an object and lift in 12.5 cm above the ground.
- *Episode End* → Success (or) 100 time-steps {Failure}

Observation Space

The following observations were considered (temporally stacked) :

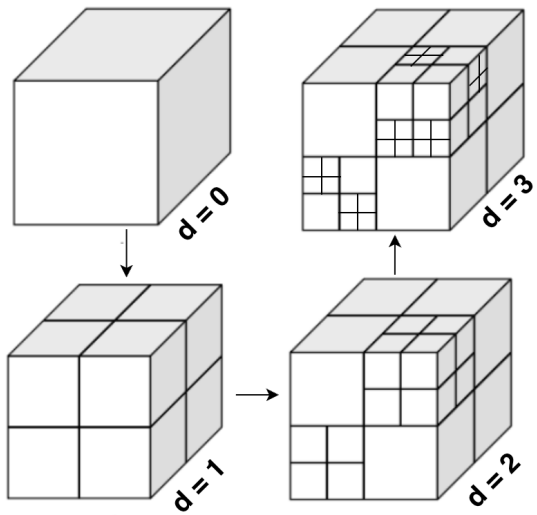
1. *Visual Octree Observations* – Data stored in the finest leaf octants is :
   - average unit normal vector, $\bar{n}$
   - average distance from centre, $\bar{d}$
   - average colour, $\overline{rgb}$



Observation Space

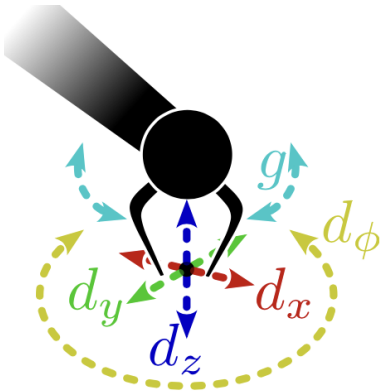
The following observations were considered (temporally stacked) :

1. *Visual Octree Observations* – Data stored in the finest leaf octants is :
   - average unit normal vector, $\bar{n}$
   - average distance from centre, $\bar{d}$
   - average colour, $\overline{rgb}$



2. *Proprioceptive Observations* – State of the gripper (open/closed) and gripper pose (position + orientation)

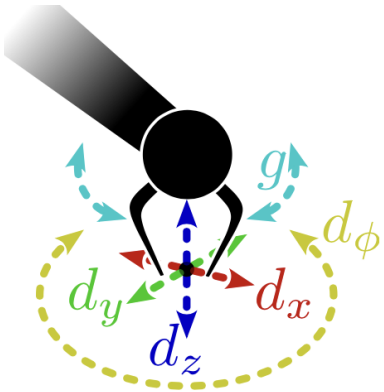
Action Space



Consists of continuous actions in cartesian space :

1. translational displacement, (d_x, d_y, d_z)
2. rotation around z-axis, d_ϕ
3. gripper closing and opening, g

Action Space



Consists of continuous actions in cartesian space :

1. translational displacement, (d_x, d_y, d_z)
2. rotation around z-axis, d_ϕ
3. gripper closing and opening, g

Reward Function

To avoid prolonged training, due to sparsity of success, via random exploration, the shown reward structure is designed to guide training.

Hierarchical Flow

Reaching	$r^0_{exp} = 1$	*(Once per episode)*
Touching	$r^1_{exp} = 7$	
Grasping	$r^2_{exp} = 49$	
Lifting	$r^3_{exp} = 343$	
Collision	$r_c = -1$	*(each timestep)*
Act Quickly	$r_a = -0.005$	

Overview of the Approach

Relevant DRL Algorithms

For this problem of *continuous control*, Model-free, Actor-Critic, Off-policy algorithms were considered.

1. **Deep Deterministic Policy Gradient (DDPG) :**[8]
   - Off-policy algorithm and makes use of Experience Replay buffer → sample-efficient.
   - Deals with continuous action spaces and trains a deterministic policy.
   - Target networks used → to ensure training stability
   - To accommodate exploration, noise is added to actions during training.

2. **Twin Delayed Deep Deterministic Policy Gradient (TD3) :** [4]
   - It is an extension to DDPG → mitigate the typical Q-value overestimation bias.
   - The following three tricks are incorporated :
     (i) Use of two individual critics where the smaller of the two is used to compute the Temporal Difference (TD) error.
     (ii) Updating actor network less often than critic networks.
     (iii) Addition of smoothing noise for the actor network

Relevant DRL Algorithms

For this problem of *continuous control*, Model-free, Actor-Critic, Off-policy algorithms were considered.

1. **Deep Deterministic Policy Gradient (DDPG) :**[8]
   - Off-policy algorithm and makes use of Experience Replay buffer → sample-efficient.
   - Deals with continuous action spaces and trains a deterministic policy.
   - Target networks used → to ensure training stability
   - To accommodate exploration, noise is added to actions during training.

2. **Twin Delayed Deep Deterministic Policy Gradient (TD3) :** [4]
   - It is an extension to DDPG → mitigate the typical Q-value overestimation bias.
   - The following three tricks are incorporated :
     (i) Use of two individual critics where the smaller of the two is used to compute the Temporal Difference (TD) error.
     (ii) Updating actor network less often than critic networks.
     (iii) Addition of smoothing noise for the actor network

Relevant DRL Algorithms

For this problem of *continuous control*, Model-free, Actor-Critic, Off-policy algorithms were considered.

1 Deep Deterministic Policy Gradient (DDPG) :[8]

- Off-policy algorithm and makes use of Experience Replay buffer → sample-efficient.
- Deals with continuous action spaces and trains a deterministic policy.
- Target networks used → to ensure training stability
- To accommodate exploration, noise is added to actions during training.

2 Twin Delayed Deep Deterministic Policy Gradient (TD3) : [4]

- It is an extension to DDPG → mitigate the typical Q-value overestimation bias.
- The following three tricks are incorporated :
  - (i) Use of two individual critics where the smaller of the two is used to compute the Temporal Difference (TD) error.
  - (ii) Updating actor network less often than critic networks.
  - (iii) Addition of smoothing noise for the actor network

3 Soft Actor Critic (SAC) : [5] (*State-of-the-Art*)

- Inherits the use of two critics and actor network smoothing noise.
- It optimizes a stochastic policy in an entropy-regularized reinforcement learning framework.
- Agent's aim to learn an effective policy by optimizing a trade-off between expected return and entropy (analogy to trading off exploration and exploitation) :

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=0}^{T} \gamma^t \left(r\left(s_t, a_t\right) + \alpha \mathcal{H}\left(\pi\left(\cdot \mid s_t\right)\right)\right)\right]$$

4 Truncated Quantile Critics (TQC) : [7] (*Mainly used here!*)

- Recently added extension to the SAC algorithm, addresses the following shortcomings :
  1. Overestimation control is coarse
  2. Taking min. of estimators (no ensemble)
- The main ideas used in this algorithm are :
  1. Models the distribution of the random return
  2. To control the overestimation, truncate the right tail of the return distribution
  3. Mixture of multiple critics is considered (Ensemble advantage).

3 Soft Actor Critic (SAC) : [5] (*State-of-the-Art*)

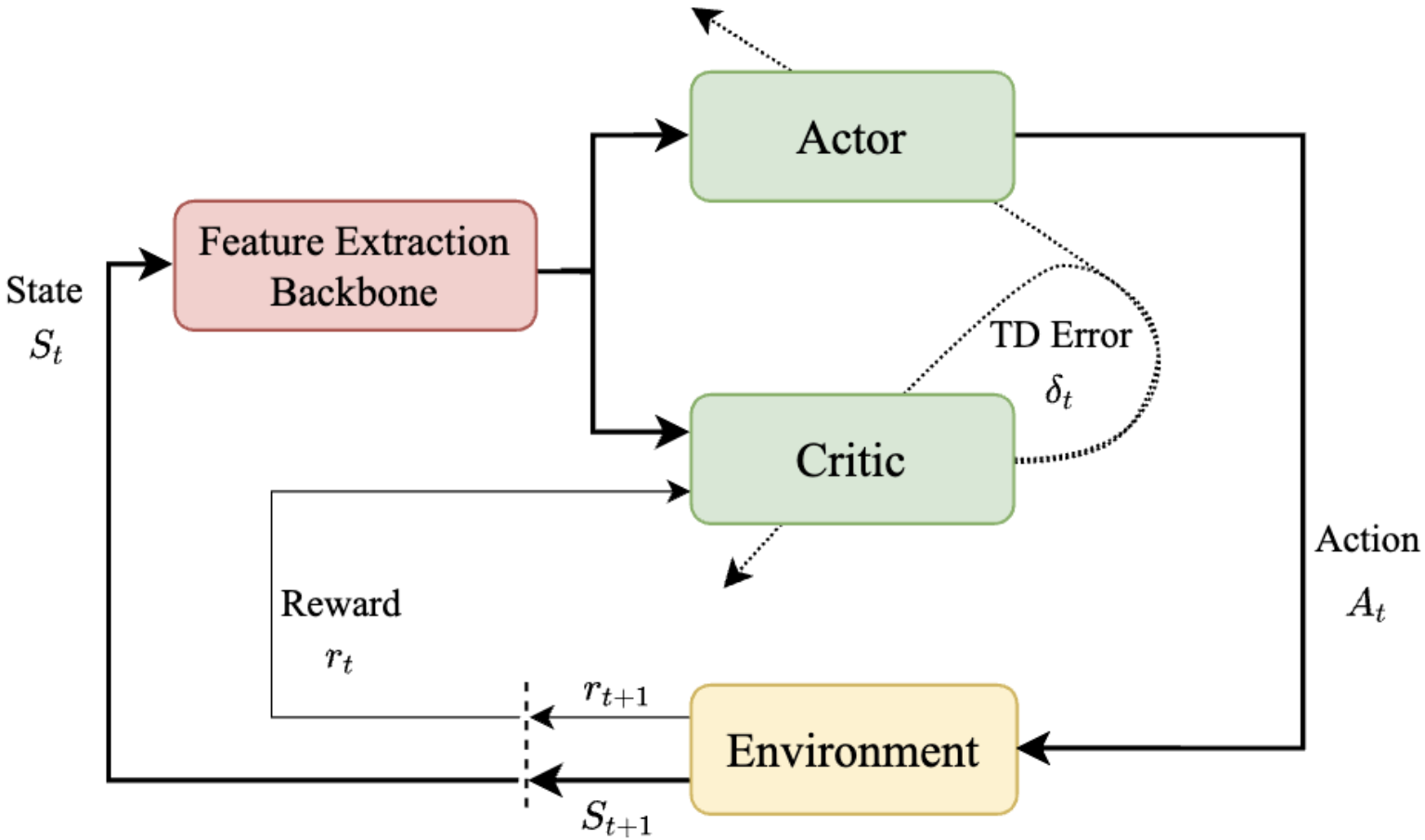
- Inherits the use of two critics and actor network smoothing noise.
- It optimizes a stochastic policy in an entropy-regularized reinforcement learning framework.
- Agent's aim to learn an effective policy by optimizing a trade-off between expected return and entropy (analogy to trading off exploration and exploitation) :

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=0}^{T} \gamma^t \left(r\left(s_t, a_t\right) + \alpha \mathcal{H}\left(\pi\left(\cdot \mid s_t\right)\right)\right)\right]$$

4 Truncated Quantile Critics (TQC) : [7] (*Mainly used here!*)

- Recently added extension to the SAC algorithm, addresses the following shortcomings :
  - (i) Overestimation control is coarse
  - (ii) Taking min. of estimators (no ensemble)
- The main ideas used in this algorithm are :
  - (i) Models the distribution of the random return
  - (ii) To control the overestimation, truncate the right tail of the return distribution
  - (iii) Mixture of multiple critics is considered (Ensemble advantage).

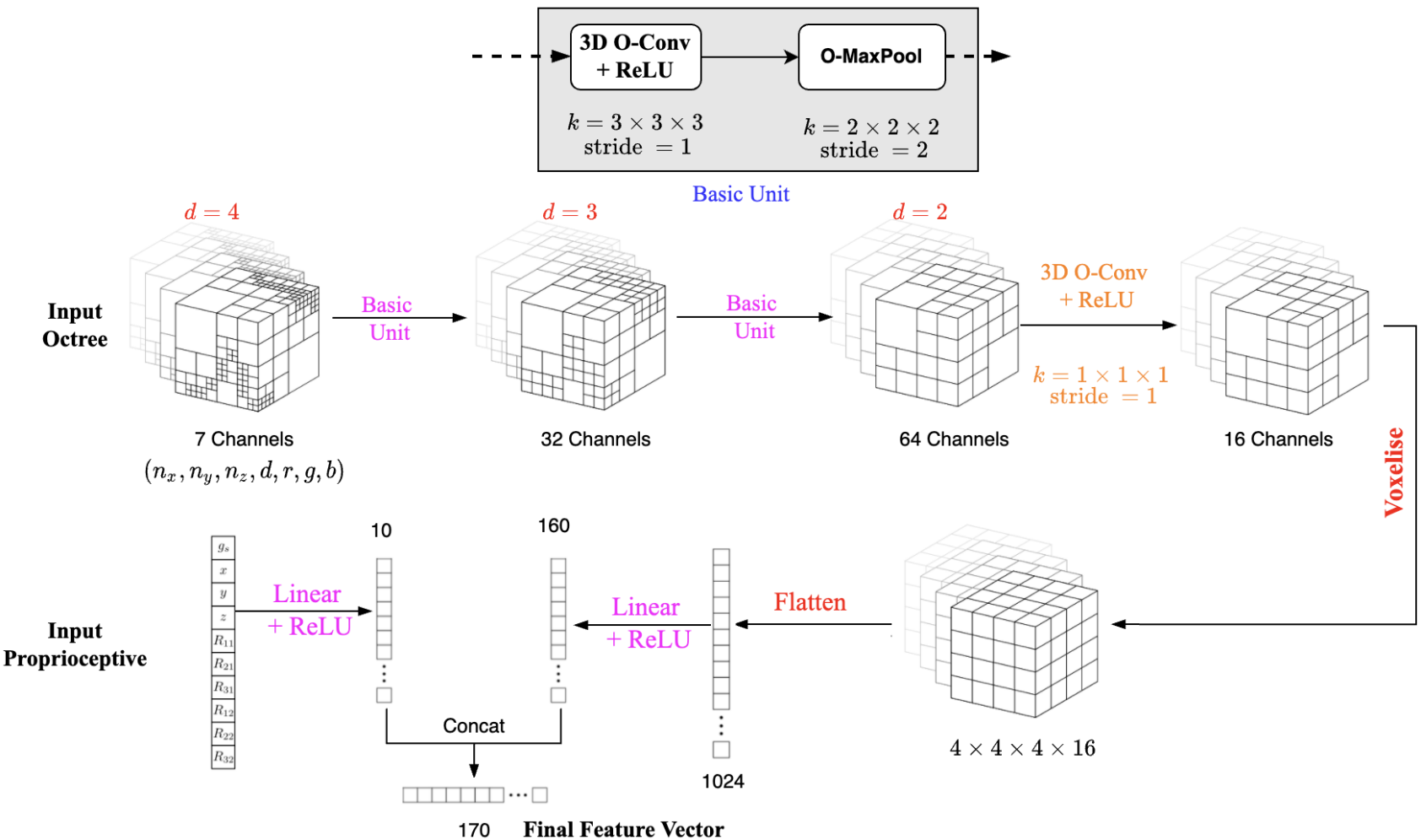
High Level Block Diagram



Design Methodology : DL Architectures

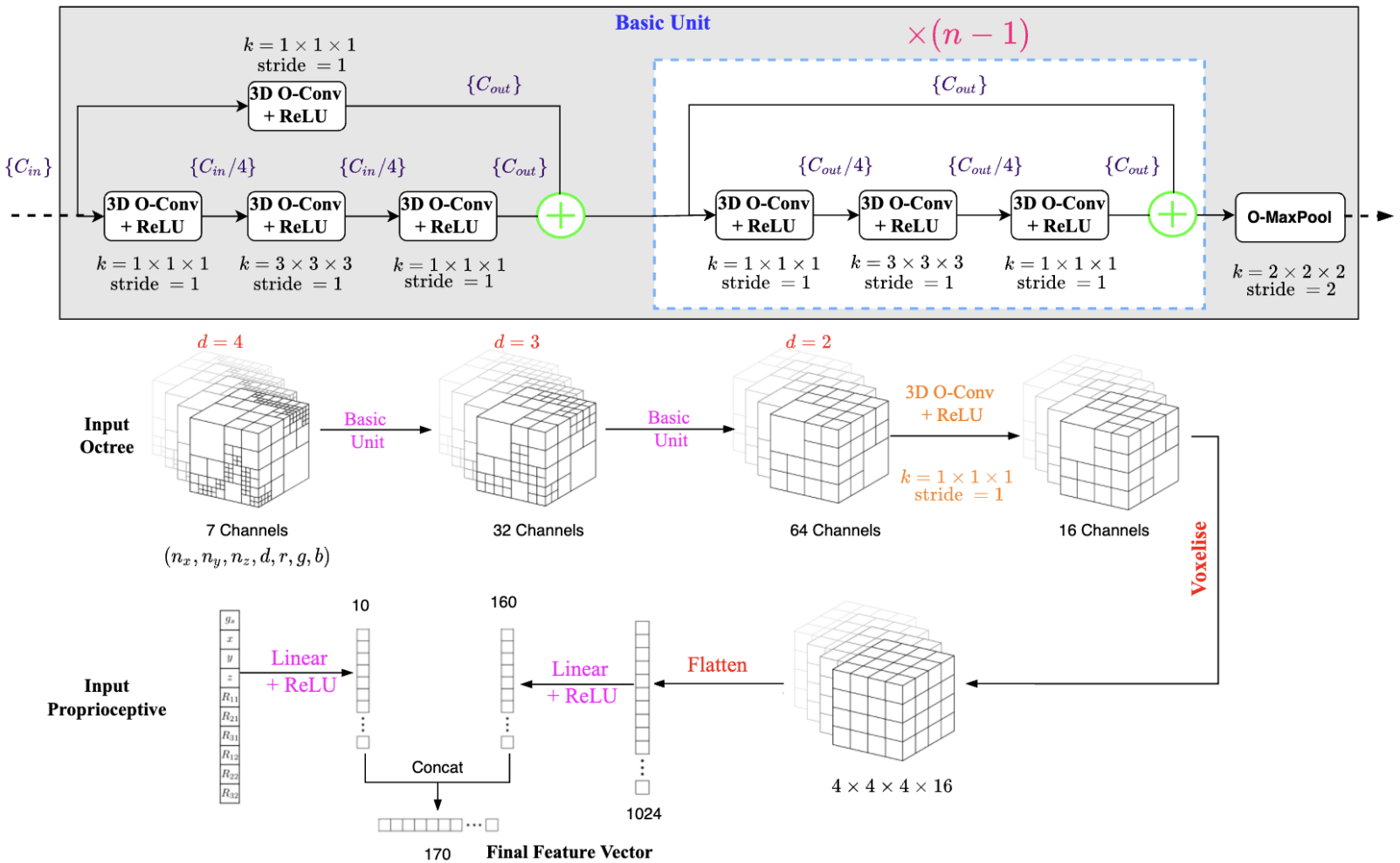
Feature Extraction Backbone

1 Vanilla O-CNN



Feature Extraction Backbone

2 Residual O-CNN

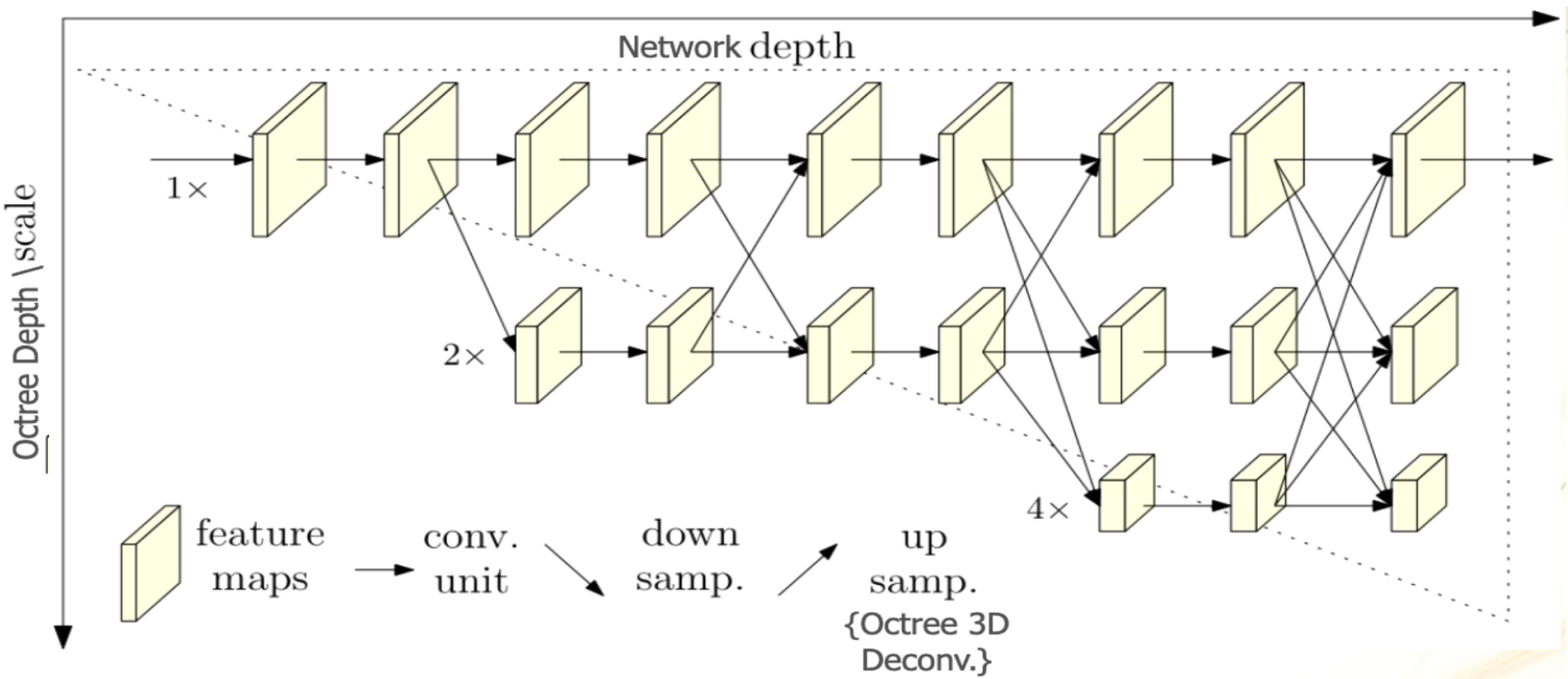


Feature Extraction Backbone

3 O-AHRNet

The main components of *O-AHRNet* are :

- Maintain **High-Resolution** octree feature representations [10]

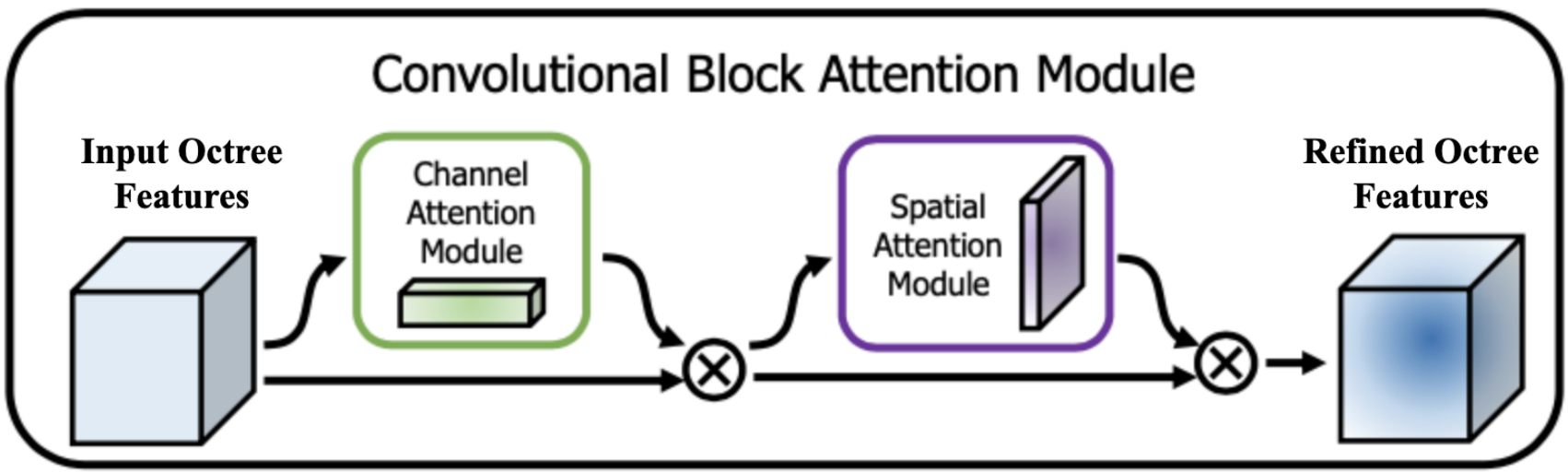


Feature Extraction Backbone

3 O-AHRNet

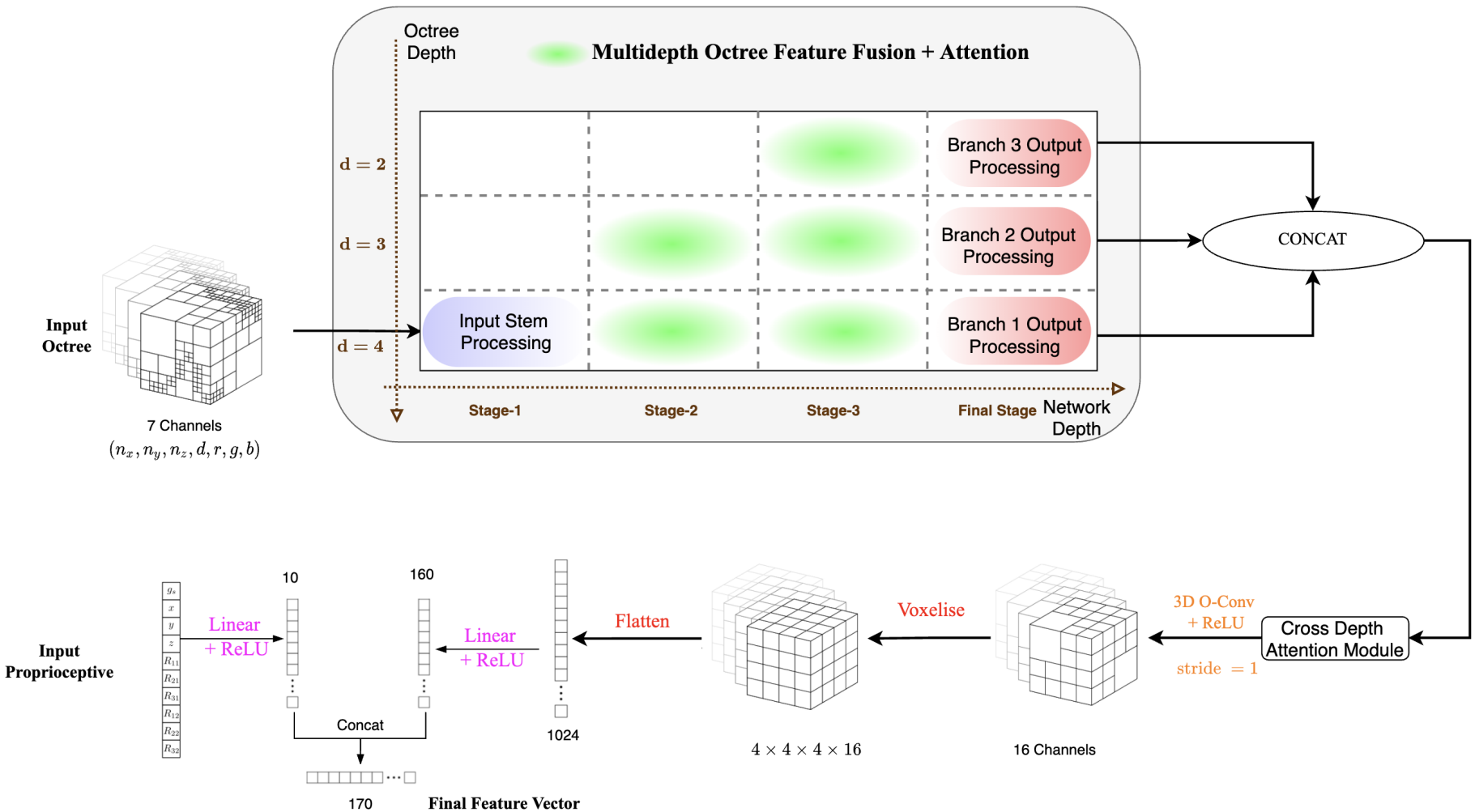
The main components of *O-AHRNet* are :

- Maintain **High-Resolution** octree feature representations [10]
- Attention Module [11][12]→ Octree convolution based efficient channel and spatial attention, using O-MaxPool & O-AvgPool features



Feature Extraction Backbone

3 O-AHRNet



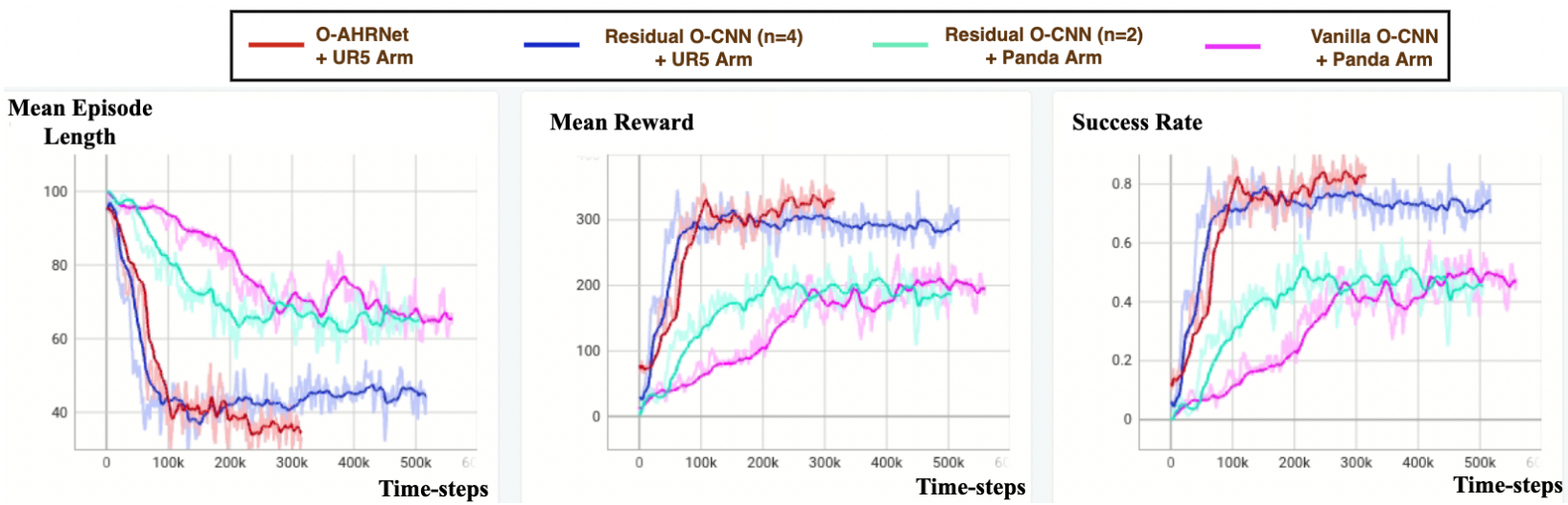
Results

Notable Results

Test (Quantitative) Results : *(200 episodes with novel objects and scenes)*

Robot	Panda	Panda	UR5 with RG2	UR5 with RG2
FE Arch.	Vanilla O-CNN	Residual O-CNN ($n = 2$)	Residual O-CNN ($n = 4$)	O-AHRNet
# Parameters	0.6795M	0.5675M	0.6348M	2.6614M
Success Rate	45.5%	45.5%	80.5%	**87.5%**
Mean Reward	188.42 ± 191.20	185.955 ± 194.76	319.37 ± 152.32	**349.34 ± 123.41**
Mean Episode Length	65.71 ± 39.38	63.965 ± 40.615	39.42 ± 34.70	**29.57 ± 31.03**
Mean Successful Episode Length	24.78 ± 19.725	20.835 ± 14.54	25.71 ± 21.95	**19.50 ± 17.04**

Training (Graphical) Curves :



Task II : Dynamic Grasping of Moving Objects

Task Description

Problem can be characterized using the following pointers:

- The target object which is moving w.r.t robot's world frame is *known*, i.e. existence and apriori access to its 3D model is assumed.
- The motion trajectory which is being followed by the target object is unknown.
- The aim of the robotic arm is to first *reach*, then *grasp* and finally *lift* the dynamic target in a stable manner.
- Grasping the dynamic target is considered to be a success if the object of interest is picked up *quickly* but without collisions.

Task Description

Problem can be characterized using the following pointers:

- The target object which is moving w.r.t robot's world frame is *known*, i.e. existence and apriori access to its 3D model is assumed.
- The motion trajectory which is being followed by the target object is unknown.
- The aim of the robotic arm is to first *reach*, then *grasp* and finally *lift* the dynamic target in a stable manner.
- Grasping the dynamic target is considered to be a success if the object of interest is picked up *quickly* but without collisions.

Task Description

Problem can be characterized using the following pointers:

- The target object which is moving w.r.t robot's world frame is *known*, i.e. existence and apriori access to its 3D model is assumed.
- The motion trajectory which is being followed by the target object is unknown.
- The aim of the robotic arm is to first *reach*, then *grasp* and finally *lift* the dynamic target in a stable manner.
- Grasping the dynamic target is considered to be a success if the object of interest is picked up *quickly* but without collisions.

Integral Components of the Approach

Integral Components of the Approach

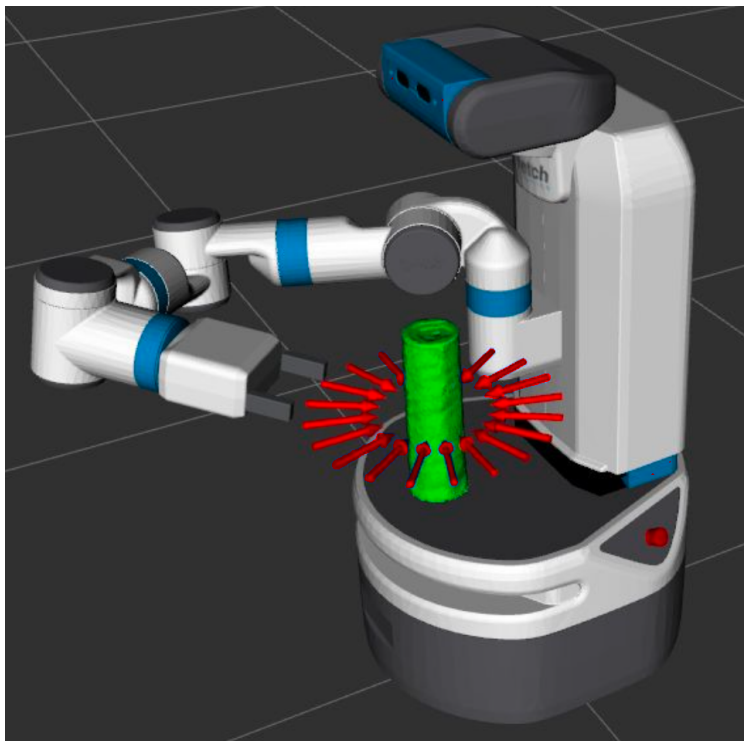
Main components of the approach are :

1. Object Pose Retrieval $\rightarrow$ Ground Truth Object Pose obtained from Pybullet Simulation Scene
2. Grasp Database
3. Motion Awareness
4. Reachability Awareness
5. Adaptive Trajectory Synthesis
6. Object Pose Prediction

Integral Components of the Approach

Main components of the approach are :

1. Object Pose Retrieval
2. Grasp Database $\rightarrow$ 5000 stable grasps generated (offline) for each object $\Rightarrow$ Top 100 reliable were chosen (to be used online)

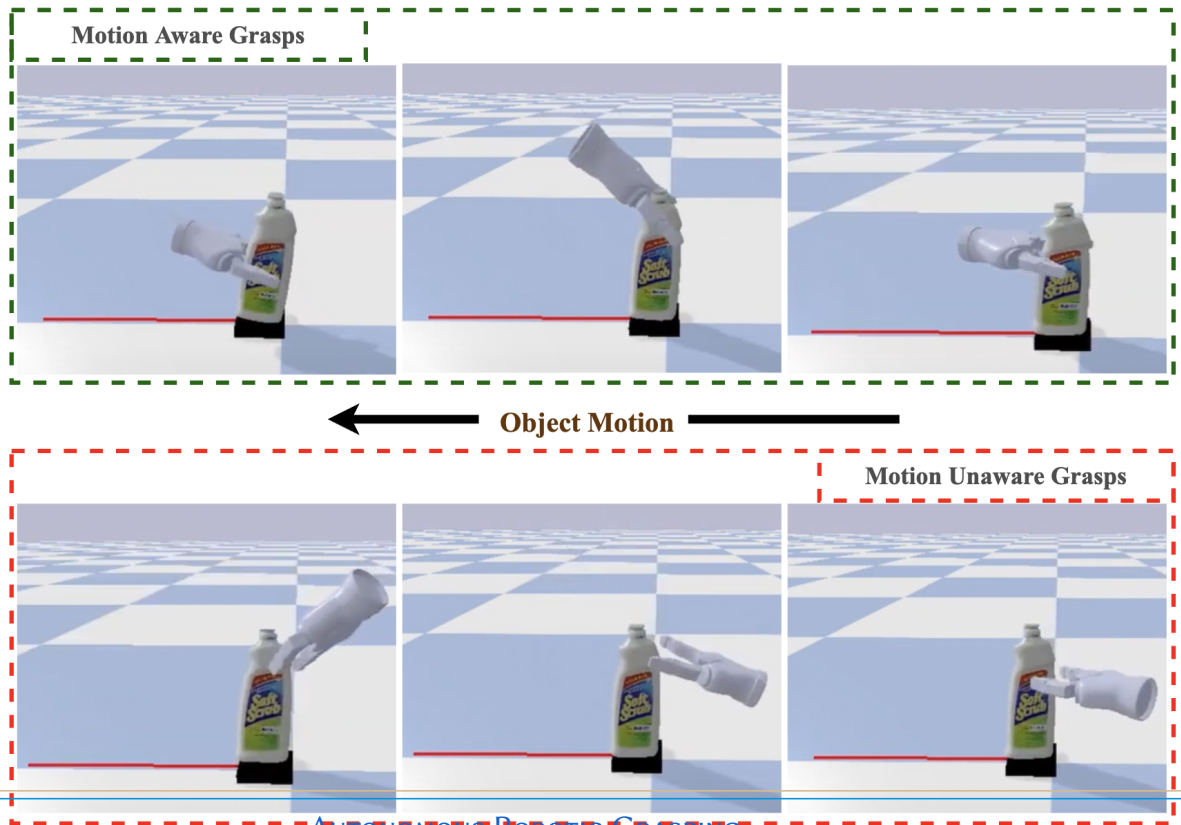


3. Motion Awareness
4. Reachability Awareness

Integral Components of the Approach

Main components of the approach are :

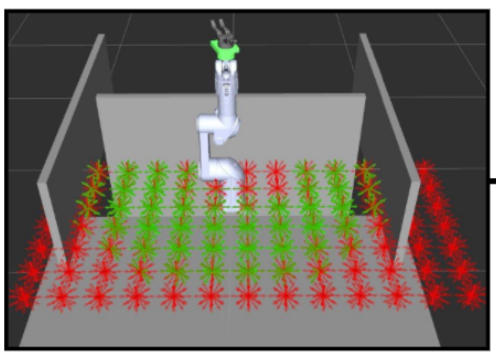
1. Object Pose Retrieval
2. Grasp Database
3. Motion Awareness $\rightarrow$ Fully Connected NN Model $\{2$ Hidden Layers$\}$ trained to predict the grasp success probability



Integral Components of the Approach

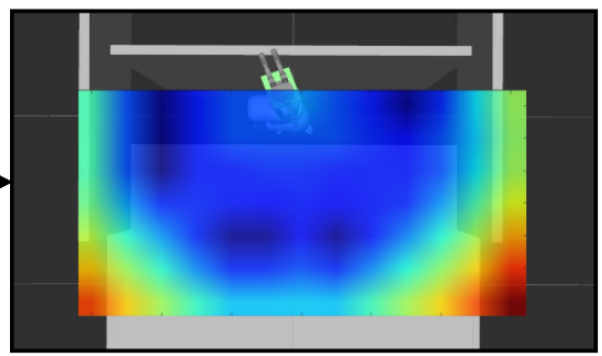
Main components of the approach are :

1. Object Pose Retrieval
2. Grasp Database
3. Motion Awareness
4. Reachability Awareness $\rightarrow$ Based on current object pose, a reachability score is computed from the reachability SDF



Algorithm 1 Reachability Space Generation

```
procedure GenerateReachabilitySpace
    poses = uniformSampleWorkspace()
    pose2Reachable = {}
    for pose in poses do
        reachable = hasCollisionFreeIK(pose)
        pose2Reachable[pose] = reachable
    SDF = computeSDF(pose2Reachable)
    return SDF
```



5. Adaptive Trajectory Synthesis
6. Object Pose Prediction

Integral Components of the Approach

Main components of the approach are :

1. Object Pose Retrieval
2. Grasp Database
3. Motion Awareness
4. Reachability Awareness
5. Adaptive Trajectory Synthesis $\rightarrow$ Trajectory seeding is done where solution from the previous timestep is used as a initial point for current trajectory
6. Object Pose Prediction

Integral Components of the Approach

Main components of the approach are :

1. Object Pose Retrieval
2. Grasp Database
3. Motion Awareness
4. Reachability Awareness
5. Adaptive Trajectory Synthesis
6. Object Pose Prediction $\rightarrow$ Necessary to estimate future object pose, since it is dynamic

Algorithmic View of the Approach

Algorithmic View (Simplified)

Algorithm 1: Dynamic Grasping Algorithm

Function *DynamicGrasp(O)*
- $G_{DB} \leftarrow RetrieveGraspDatabase(O)$;
- **while** *True* **do**
  - $p_c \leftarrow RetrieveObjectPose(O)$;
  - $t_{pred} \leftarrow CalcPredTime(d)$;
  - $p_f \leftarrow PredictObjectPose(O, t_{pred})$;
  - $G_W \leftarrow ConvertGrasps(G_{DB}, p_f)$;
  - $G_F \leftarrow FilterGrasps(G_W, p_f)$; `/* Using Ranking Functions */`
  - $g_c \leftarrow ChooseGrasp(G_F)$;
  - If this chosen grasp is unreachable, continue to next iteration;
  - Move arm to g_c;
  - **if** *ObjectGrasp()* **then**
    - $p'_c \leftarrow RetrieveObjectPose(O)$;
    - $t'_{pred} \leftarrow 1s$;
    - $p'_f \leftarrow PredictObjectPose(O, t'_{pred})$;
    - $g_c \leftarrow ConvertGrasps(G_{DB}, p'_f)$;
    - Move arm to g_c;
    - Close hand while moving with the target for t'';
    - Break Loop;
  - **end**
- **end**
- **return** CheckGraspSuccess();

end

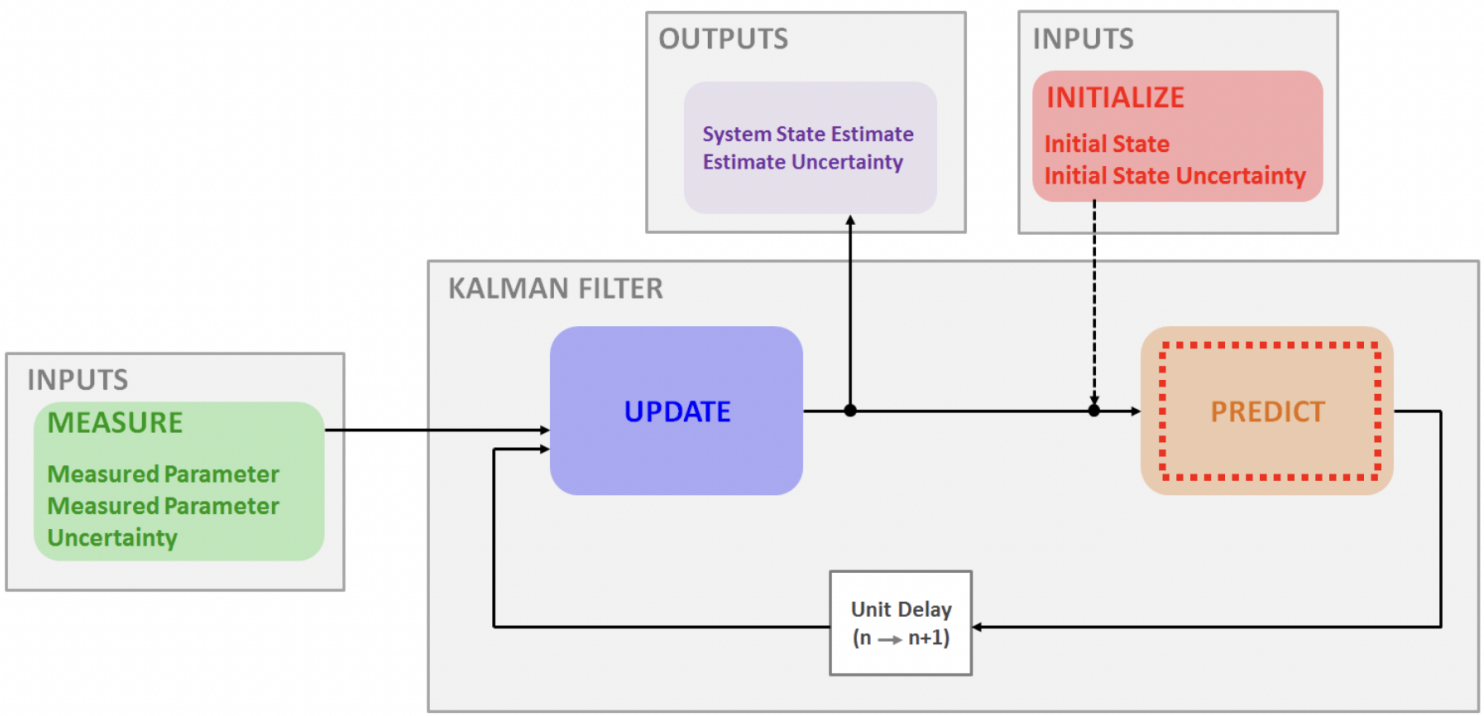
Object Pose Prediction : Method Design

Object Pose Prediction

1 Kalman Filter

Two important steps are :

- **Predict Step** → Forecast the future state using Constant Acceleration Dynamical Model.
- **Update Step** → Correct the prediction after reception of the sensor measurement.

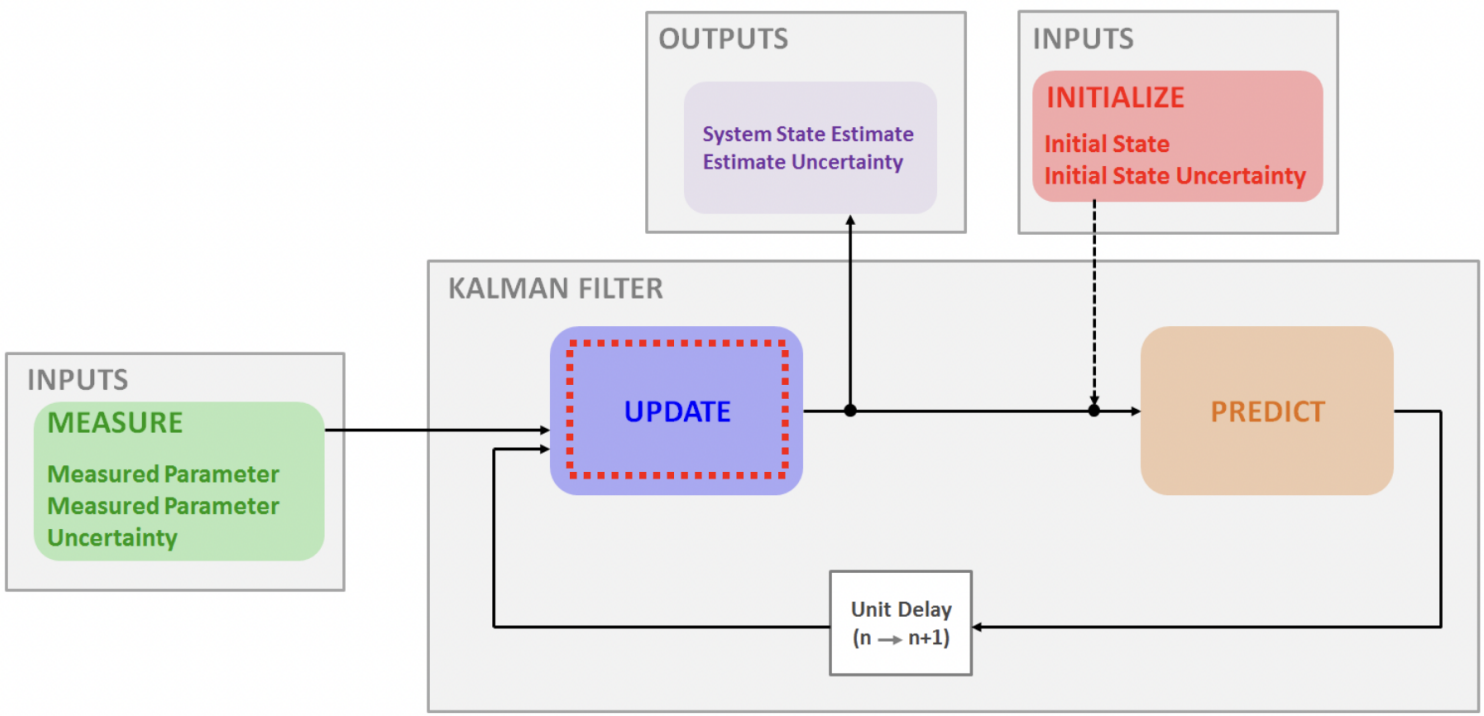


Object Pose Prediction

1 Kalman Filter

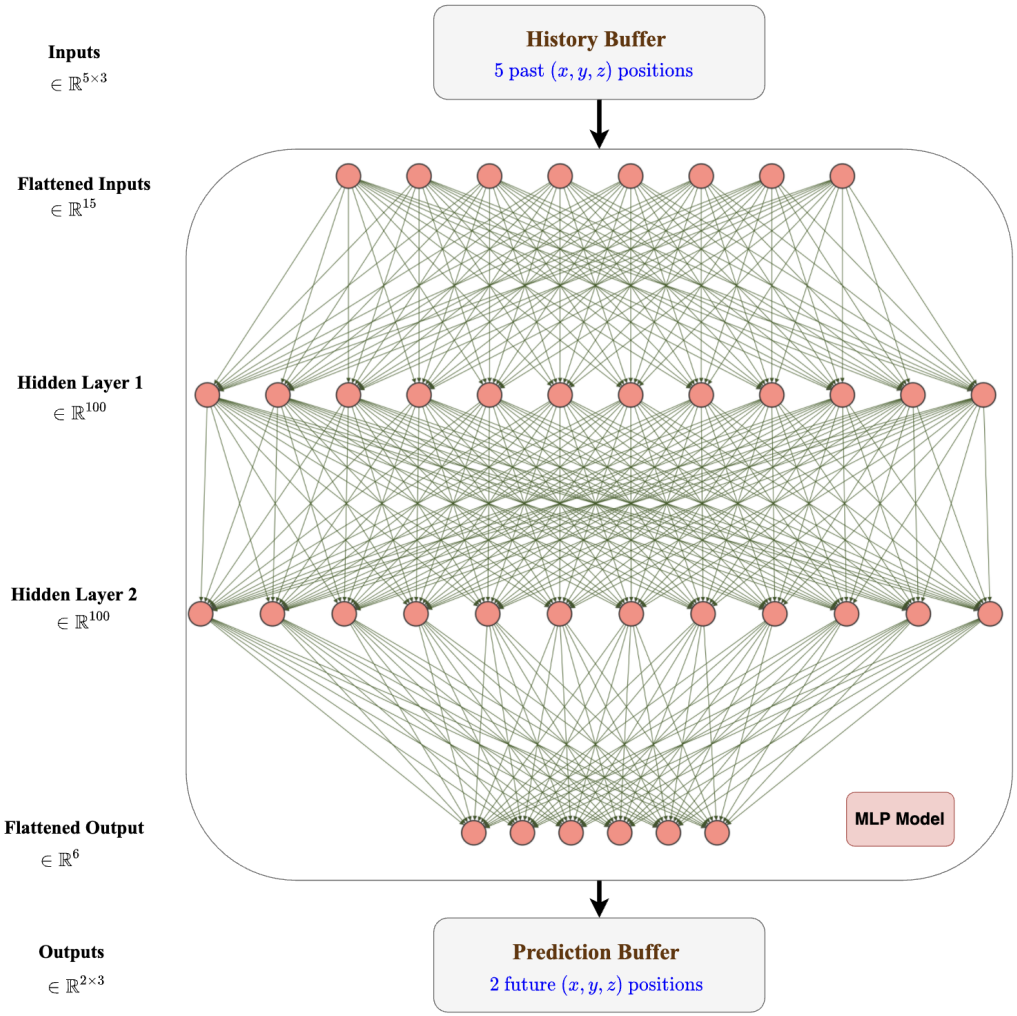
Two important steps are :

- **Predict Step** → Forecast the future state using Constant Acceleration Dynamical Model.
- **Update Step** → Correct the prediction after reception of the sensor measurement.



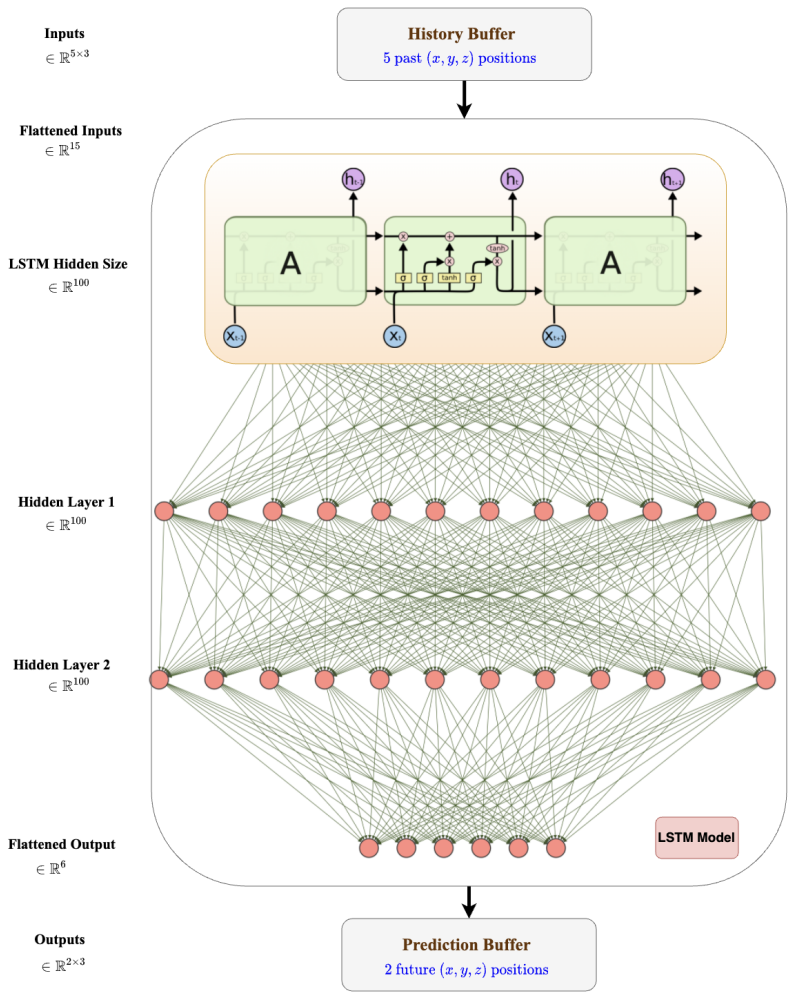
Object Pose Prediction

2 Multi Layer Perceptron (MLP)



Object Pose Prediction

3 Long Short Term Memory (LSTM) Networks



Results

Notable Results

Test Setup → 100 test trials are conducted on the following objects :

Power Drill | Bleach Cleanser | Mustard Bottle | Potted Meat Can | Sugar Box

Method \ Type of Motion	Linear	Linear with Obstacles	Linear with Top Shelf	Sinusoidal
Kalman Filter	**85.8**	82.8	46.6	10.8
	13.7016	**14.6598**	25.0018	24.0029
Multi Layer Perceptron (MLP)	85	81.2	46.2	74.8
	14.7262	15.0821	25.0037	18.0111
Long Short Term Memory Network (LSTM)	85.4	**84.4**	**46.2**	**75.8**
	13.9090	14.7954	**24.5845**	**17.8904**

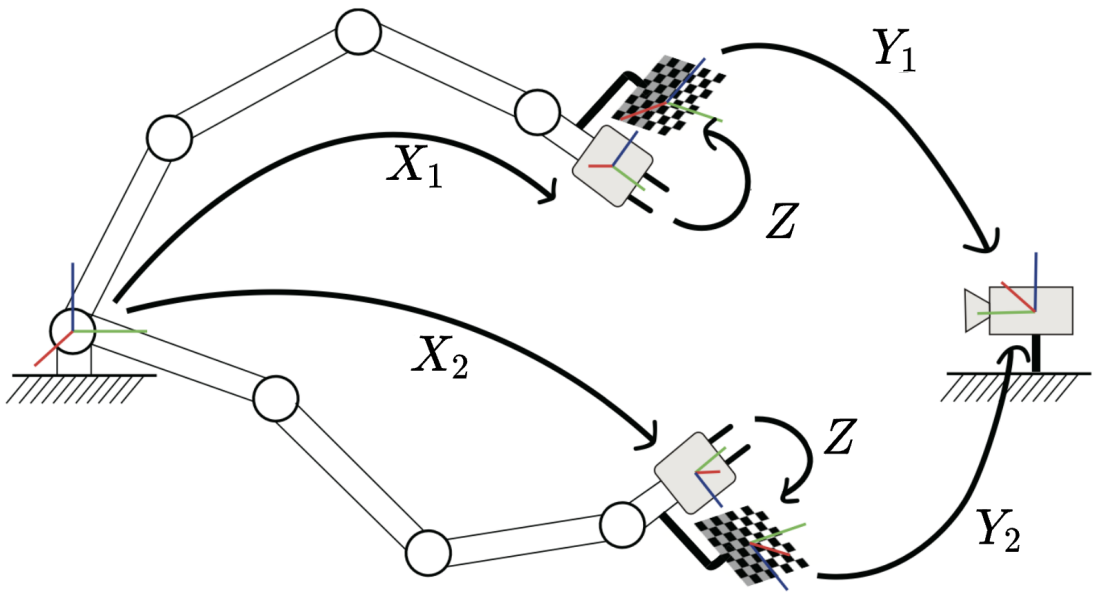
■ Average Success Rate || ■ Average Dynamic Grasping Time

Real Robotic Setup

Real Robotic Setup

The following two subtasks were considered :

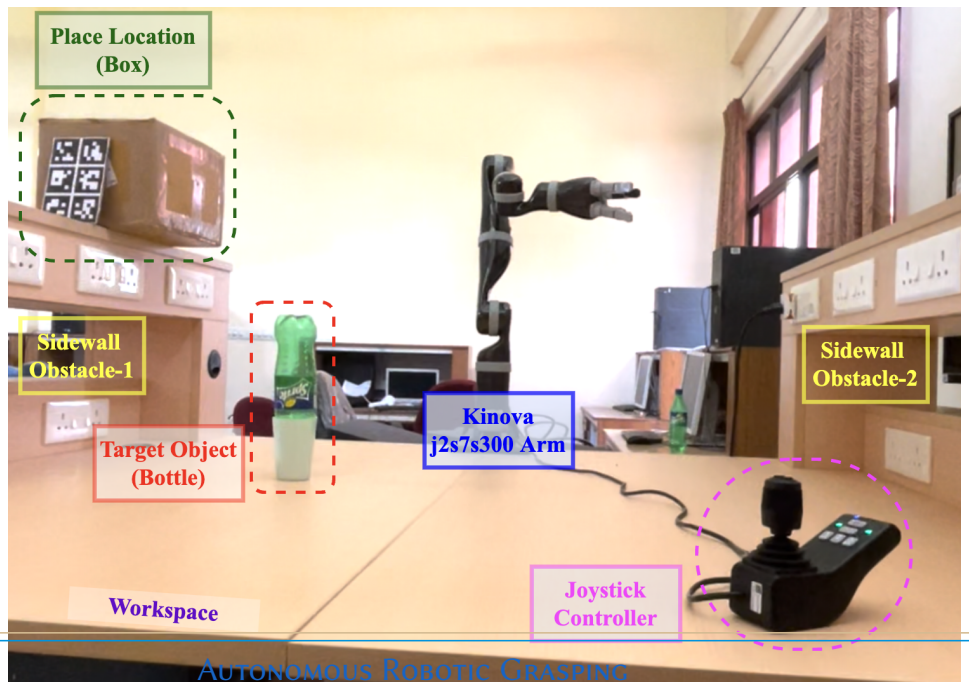
- **Hand-Eye Calibration** $\rightarrow$ To utilise visual input, we must determine the transformation between robot's end effector and camera coordinate systems.
- **Blind Pick and Place** $\rightarrow$ *Pick* an object of interest from a particular location and *Place* it at the desired location. Locations are predetermined.



Real Robotic Setup

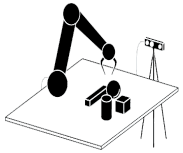
The following two subtasks were considered :

- **Hand-Eye Calibration** → To utilise visual input, we must determine the transformation between robot's end effector and camera coordinate systems.
- **Blind Pick and Place** → *Pick* an object of interest from a particular location and *Place* it at the desired location. Locations are predetermined.



Demonstration (Prerecorded)

A Glimpse of the Test



Thank you !

References I

[1] Iretiayo Akinola, Jacob Varley, Boyuan Chen, and Peter K Allen. "Workspace aware online grasp planning". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pages 2917–2924.

[2] Iretiayo Akinola, Jingxi Xu, Shuran Song, and Peter K Allen. "Dynamic Grasping with Reachability and Motion Awareness". In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pages 9422–9429.

[3] Konstantinos Daniilidis. "Hand-eye calibration using dual quaternions". In: *The International Journal of Robotics Research* 18.3 (1999), pages 286–298.

[4] Scott Fujimoto, Herke Hoof, and David Meger. "Addressing function approximation error in actor-critic methods". In: *International conference on machine learning*. PMLR. 2018, pages 1587–1596.

[5] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. "Soft actor-critic algorithms and applications". In: *arXiv preprint arXiv:1812.05905* (2018).

[6] Jiawei Hou, Yizheng Zhang, Andre Rosendo, and Soren Schwertfeger. *Mobile Manipulation Tutorial*. 2020.

[7] Arsenii Kuznetsov, Pavel Shvechikov, Alexander Grishin, and Dmitry Vetrov. "Controlling overestimation bias with truncated mixture of continuous distributional quantile critics". In: *International Conference on Machine Learning*. PMLR. 2020, pages 5556–5566.

[8] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. "Continuous control with deep reinforcement learning". In: *arXiv preprint arXiv:1509.02971* (2015).

[9] Andrej Orsula. "Deep Reinforcement Learning for Robotic Grasping from Octrees". Master's thesis. Aalborg University, 2021.

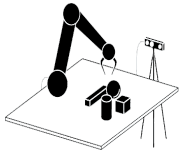
[10] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. "Deep high-resolution representation learning for human pose estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pages 5693–5703.

References II

[11] Qilong Wang, Banggu Wu, Pengfei Zhu, P. Li, Wangmeng Zuo, and Qinghua Hu. "ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020), pages 11531–11539.

[12] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. "Cbam: Convolutional block attention module". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pages 3–19.

[13] Alex Becker (www.kalmanfilter.net). *Online Kalman Filter Tutorial*.



Questions?