72th International Astronautical Congress, Dubai, United Arab Emirates, 25-29 October 2021.

IAC–21–D1.6.7

TOWARDS INCREMENTAL AUTONOMY FRAMEWORK FOR ON-ORBIT VISION-BASED GRASPING

Kuldeep R Barad[a b *], Carol Martinez [a], Jan Dentler[b], Miguel Angel Olivares Mendez[a]

[a] Space Robotics Research Group, SnT-University of Luxembourg, 29 Ave. John F. Kennedy, 1855 Luxembourg
[b] Redwire Space Europe, 5 Rue de l'Industrie, 1811 Luxembourg
[*] Corresponding author, email: kuldeep.barad@uni.lu

Abstract

This work presents a software-oriented autonomy framework that enables the incremental development of high robotic autonomy. The autonomy infrastructure in space applications is often cost-driven and built for a narrow time/complexity domain. In domains like On-orbit Servicing Assembly and Manufacturing (OSAM), this prevents scalability and generalizability, motivating a more consistent approach for the incremental development of robotic autonomy. For this purpose, the problem of vision-based grasping is described as a building block for high autonomy of dexterous space robots. Subsequently, the need for a framework is highlighted to enable bottom-up development of general autonomy with vision-based grasping as the starting point. The preliminary framework presented here comprises three components. First, an autonomy level classification provides a clear description of the autonomous behavior of the system. The stack abstraction provides a general classification of the development layers. Finally, the generic execution architecture condenses the flow of translating a high-level task description into real-world sense-plan-act routines. Overall, this work lays down foundational elements towards development of general robotic autonomy for scalablity in space application domains like OSAM.

keywords: OSAM, Autonomy, On-orbit Robotics, Perception, Manipulation, Learning

1. Introduction

Advances in On-orbit Servicing, Assembly and Manufacturing (OSAM) developments show a clear trend in decreasing mission costs by using scalable robotic technologies and enabling more autonomous operations.[1] The use of robotic manipulators have been ubiquitous in the OSAM context and will remain so following the trend low-cost robotic arms for spacecraft.[2] Still, the existing robotic systems for spacecraft[3–5] rely on conventional hardware, which involve high cost of development and conventional algorithms that have very low generalizability in the broad OSAM application space. This motivates the development of modular and autonomous vision-based manipulation systems that enable cost-effective and scalable OSAM infrastructures.

A foundational element in enabling highly autonomous operations with dexterous space robotics systems is the intelligent visual perception subsystem that can estimate dynamic grasps for unknown objects in space. The problem of interest is the use of visual information to capture and manipulation of unknown objects present in host spacecraft's workspace. These target objects may be tumbling and their physical and visual properties (material, geometry and state) are partially or totally unknown. The importance of visual perception is due to its ability to capture dense information and consequently, the context about the environment in real time. In recent times, the motivation to use visual perception on-board has been complemented by 1) availability of commercial sensors and camera systems that have lower form-factors, cost, power and hardware complexity, and 2) the rapid developments in data-driven algorithms for unstructured and uncertain environments. Despite the promise of vision systems, there are unique challenges to a space-borne vision system. The space-borne vision sensors suffer from high signal-to-noise ratio, rapidly varying brightness and background contrast as well as ineffectiveness during eclipse. Additionally, extracting meaningful high-level information such as semantic features in the images from pixel intensities cannot often be accomplished through analytical methods. A visual perception system that overcomes these challenges and reliably extract meaningful information can improve embodiment of space-borne manipulation systems to enable increasingly complex use cases. With improving intelligence and embodiment, the systems can be made increasingly autonomous. Autonomy of a system is its ability to achieve goals while operating independently of external control.[1] In space systems particularly, increasingly autonomous systems enable time-critical decision making in the presence of time delays or communication losses. The online decision making also enables the use of rich

on-board sensor data instead of bandwidth limited downlink data. Further, autonomy eliminates the dependency of system performance on human factors' and reduces the cost of related ground support infrastructure. As the demand for space infrastructure sustainability and scalability grows, autonomy stands to enable cost and risk reduction with dramatic improvement in overall productivity of space systems.

This work focuses on the two elements highlighted above. First, the problem of vision-based grasping as a fundamental visuomotor skill for an intelligent and dexterous space robot is described. Then, the question of general autonomy development is addressed using a software-oriented autonomy framework, within which robot capabilities may be incrementally and consistently developed. To this extent, Section 2 lays down a brief assessment of the vision-based grasping problem, one of the fundamental visuomotor skills that can compositionally enable general-purpose space robots. Subsequently, Section 3 attempts to lay down the formalism of general autonomy. A preliminary autonomy framework is presented within which vision-based grasping and other visuomotor skills can be developed to incrementally enable high autonomy in OSAM as well as general large-scale space robotic applications.

1.1 Related Work

General and reusable software architecture for terrestrial and space robotics has been an actively evolving topic for research. The aim is to decompose a robot task and into independent sub-problems to flexibility in recomposing them to produce flexible task-level behavior.[6] The nature of decomposition has varied, starting from Brooks et al.[7] where task is decomposed into levels of competences and corresponding layers of control. Instead of decomposing the problem into perception, planning, mapping, control and other functional modules, the architecture provides a mobile robot with seven general competences. These include obstacle avoidance, explore aimlessly, map the environment, register dynamic and static elements et cetera. Bonasso et al.[8] devised a three tier decomposition of robot control into reactive skills, sequencing and deliberation. Volpe et al.[9] proposed CLARATy arguing the improvements provided by a coupled two layer architecture that tightly integrates planning and execution. The bottom layer is a functional layer that interfaces with the hardware, while the decision layer at the top decomposes higher-level goals into sub-goals and schedules them. Brat et al.[10] propose a highly compositional architecture for space applications by combining object-oriented functional layer of CARATy[9] with a constraint-based EUROPA[11] planning layer, using PLEXIL.[12] Closer to this work, ERGO[13] provides an autonomy framework based on T-REX,[14] for development of robotic operation at various autonomy levels. Another line of work in robotic architectures have focused more on the lower level implementation to enable real-time operation and interface standardization.[15,16] Recent advances in robot learning often devise end-to-end architectures[17,18] that focus on learning visuomotor representations that translate goals into sequential real-world actions.

Along this line of work, we present the autonomy framework in the context of a consistent bottom-up development of primary skill- vision-based grasping. The preliminary framework comprises three components. First and foremost, autonomous behavior description is outlined using a systematic autonomy level classification. The framework inherits the notion of decomposition from conventional robotic architectures at three capability levels-task, subtask, and skill. A flexible execution architecture brings these components together to represent generic execution for any autonomy level. The architecture follows a decomposition and re-composition flow for the three capability levels to execute real-world goal-oriented actions. To support the execution flow, we introduce the abstraction of the autonomy stack. Autonomy stack inherits from the trends in the use of learning-based methods and ontology-based knowledge representations used in contemporary terrestrial robotic systems.

2. Vision-based Grasping for OSAM

Traditionally, space-borne robotic manipulators were most often used for docking, berthing and assistive assembly/repair in space in orbit.[19–21] These were typically accomplished under the assumption of target fiducials, accurate model knowledge, mission-specific grappling fixtures and end-effectors. In realizing the potential of OSAM activities at scale, the mission-specific fiducials, models and end-effectors greatly limit the application space by imposing limitations on the kind of objects, environments and scenarios that can be dealt with by the special purpose manipulator. To push the utility of space manipulators forward and towards general-purpose manipulation systems for OSAM (and planetary applications), manipulator systems need to be able to perceive and handle objects of various shapes, sizes and properties in dynamic and uncertain on-orbit scenarios.

2.1 Problem Formulation

The problem under consideration is the stable grasp of an object in space. The capability to accurately and reliably grasp objects in space is at the core of servicing, assembly and manufacturing in orbital as well as planetary context. Their scalability depend greatly on the ability to

work autonomously in unstructured and uncertain environments [*].In context of space-borne and especially on-orbit manipulators, the micro-gravity adds a dynamic component which imposes high accuracy of grasp configuration estimation along with high precision of alignment and grasp execution. The contact dynamics are more sensitive to contact interface design unlike terrestrial applications where often restricting (bottom) plane is used to aid success in imprecise grasping. Further, the possibility of damage or out-of-workspace drift imposes more stringent requirement on grasp generation and execution.

The problem formulation has one more axis of variation once the visual modality (RGB or RGB-D) and the end-effector type (n-fingered gripper) are fixed. This is the apriori knowledge of the target object. As outlined in Bohg et al.,[22] this aspect can be classified into 3 categories:

1. **Known Objects**: Model and interaction properties are known perfectly and pre-computed grasps can be used to select a suitable grasp that complies with the overarching task, once the object pose is estimated from the image.

2. **Familiar Objects**: Model and interaction properties may not be known, but the target is similar to an object encountered during offline model/database/experience generation. Similarity can be in terms of shape, size, texture, features, or an arbitrary high level category that allow grasp synthesis and selection to be adapted based on appropriate object representations and similarity metrics.

3. **Unknown Objects**: No assumption is made on models or experience. Grasp synthesis and selection must rely on features or structure in real-time sensor data.

Further assumptions are made on the dynamic nature of the target object during pre-grasp period, in order to decouple the manipulation and grasping problem from the 6-DOF spacecraft control. The target object is assumed to be free-floating before grasping, but the residual momentum of the potentially tumbling target relative to the servicing spacecraft is assumed to be negligible or within compensation margins of on-board reaction wheels or compliance control mechanism. This excludes cases of grasping large bodies with high relative momentum at grasp time. This is justified by the fact that large relative momentum can be negated by achieving motion synchronization relative to such a body, as conceived in debris capture architectures,[5] using servicer GNC without coupling it with the manipulation problem, and therefore suitably ignoring it.

[*]The notion of environment here also includes secondary objects and agents

2.2 Problem Breakdown

The problem of taking an image or sequence of images and producing actions that result in a stable and task-compliant grasp can be decomposed broadly into grasp synthesis, motion planning, grasp tracking and grasp execution. Most commonly, the methods presume an ROI that may be produce using object detection or scene segmentation.

Grasp synthesis concerns the problem of finding a grasp configuration that satisfies a set of criteria regarding one of more of- dexterity, equilibrium, stability and specific dynamic behaviour.[23] Grasp configurations are parameterized generally using a reference 3D grasp point on the object, approach axis along which the fingers close-in and the initial gripper configuration, and optionally the arm and wrist pose that comply with a task. While exact representations vary,[24,25] they directly or indirectly encode the aformentioned parameters for parallel jaw grippers, which can be generalized for grippers with more than two fingers and non-parallel layouts.[26] Given the representation, the persistent problem in grasp synthesis is for the algorithms to sample discrete grasps from an infinite set of possible grasp configurations, rank and select a task compliant grasp as per a grasp quality metric.[27,28] Motion planning then concerns to the problem of planning the optimal approach and grasp using the kinematic and dynamic models of the manipulator, and grasp execution concerns the generation of real-time actuator commands in presence of uncertainty in perceptual input, object representation like pose and shape, friction and actuator dynamics. Finally, grasp tracking concerns the update of grasp configuration in presence of relative dynamics between the arm and the target object.

2.3 Solution Approaches

The solution approaches to the vision-based grasping problem vary greatly, but in the last decade generally tend towards augmenting or replacing the traditionally used analytical approaches with data-driven methods. This is motivated by the fact that purely analytical methods, require knowledge of some or all of- geometric, kinematic and dynamic properties of the object apriori. Moreover, the computational effort necessary for optimizing over grasp criteria to find a task-oriented grasp configuration is intractable for arbitrary new objects.[23] Empirical or data-driven approaches reduce the computational complexity, by using pre-computed grasps from an on-board knowledge base or by sampling limited grasps using a heuristic or a learned mapping.

For known objects, the problem simplifies to that of object detection and pose estimation, after which a pre-computed grasp can be executed subject to reachability

and task constraints. The pre-computed grasps are generated from either human/expert demonstration[29,30] or offline grasp synthesis algorithms and ranking metrics.[31–33] Probabilistic frameworks are a common choice when using 3D observations for detection and pose estimation. They are based on part-hierarchy object models as in Detry et al.[34] or fit observed point cloud to a known point cloud model using iterative closest point algorithm. For cases where monocular images are used, it is now common to use mature CNN architectures. The use of machine learning in general and CNNs in particular has also been motivated in recent works on uncooperative rendezvous.[35,36] CNNs show effectiveness in tackling strong variations in illumination conditions and presence of adverse visual artefacts like light, shadow and earth background. In principle, CNNs can be used to detect target and regress 6D pose directly as in Sharma et al.[36] Alternatively after the target detection, a keypoint detection network[37] maps image to pre-defined keypoints of the known 3D model. Using correspondences of detected keypoints and 3D model, the pose is then estimated using a Perspective-n-Points solver. In comparison to learned models that directly estimate 6D pose, approaches that use keypoints and correspondence have shown superior pose estimation accuracy[38] and to an extent also enable explainability resulting from intermediate outputs and confidences.

For familiar objects, grasp synthesis from a segmented object image is based on the assumption that successful grasps for similar objects can be generated in a similar way. The primary problem is to select similarity representation that can be decoded from a visual input. Subsequently, a similarity metric is necessary to express whether previous grasp experience may generalize and generate a stable grasp, subject to reachability constraints. It is common to use affordances that represent functionality of the target object[39] to encode similarity in terms of grasping. It is most common to use learning to encode high-level affordance representations as well as to generalize over familiar yet unseen objects.[22] Many works utilize supervised learning to encode a mapping that either classifies and qualifies a grasp candidate in a dicriminative manner, or regresses a grasp configuration directly in a generative manner. For both types of methods, scene segmentation is usually necessary to recognize and isolate the object. Discriminative approaches often use CNNs to map a grasp configuration to grasp quality or a predictive success criteria using visual features in the segmented image.[40–42] Generative techniques estimate one or more suitable grasp configuration by direct regression or classification in dicretized configuration space.[25,43,44] While many of these approaches that use supervised learning show successful adaptation of the grasp synthesis on novel objects, the generalization is achieved within a distribution that generates the training data. Consequently, the grasp synthesis approach can be deemed applicable only for the objects that are similar, subject to the learned features. These approaches eliminate the need for reconstruction or correspondence-based pose estimation, as 3D model knowledge is not assumed. Many approaches that use learning show generalization, which for practical purposes can handle grasp synthesis for unknown object sufficiently well. It is ambiguous whether the approaches that use model-free learning and generalize well over unseen objects can be considered to handle unknown objects. Especially when using exploration and domain randomization during learning, learnt policies are able to generalize over large range of unseen objects in various environments.

According to the classification in Bohg et al.,[22] experience similarity might not be considered to solve object grasping. In that case, only heuristic based methods that use structure extraction using (partial) shape reconstruction.[45–47] It is noted that most approaches mentioned in this section involve assumption of a restricting plane at the bottom as in table-top grasping. Often, the implementations are open-loop which makes them susceptible to the noise in perception input. Few works have explored on closed-loop 6DoF grasping[43,48–50] that can extended well for space application.

2.4 Generalizable Manipulation Autonomy

The ultimate goal for high autonomy systems is to enable sequential decision making in complex environments. This implies that the robots will need to execute tasks in a goal oriented manner while handling and encountering a broad range of objects, scenes, semantics and task goals along its life-cycle. Traditionally, model-based methods have introduced increasing degree of generalization in perception, planning and control. However, it is usually constrained to narrow operating domains in structured environments. This is clearly limited by availability and development of models, metrics and representations for inherent uncertainty in the environment. Therefore, robot learning has been considered an integral element in enabling practical generalization.[51] However, recent advances in machine learning and data-driven algorithm design have shown rapid progress in computer vision and natural language tasks like object recognition[52] and scene description.[53] It has provides the precursors for developing modern building blocks for generalizable robotic autonomy that works across high complexity tasks and dynamic environments.

Recent work on this front has focused on multiple levels from the low-level skill generalization to task structure generalization and concept discovery. These approaches

use reinforcement learning which allows exploration of the relevant state-action domain. At a skill level, generalization can be improved by choosing the right state and/or action spaces. Most recently, Martin et al.[54] show that using a known model of a robot arm, choosing pose, velocities and impedance gains as action space variables instead of actuator commands improves generality of learned manipulation skill. They also show direct transfer of skill policy from simulated training to real-world operation on different robots. Allshire et al.[55] go further by eliminating the need to tailor the action space and instead automatically learn a latent action-space relevant for a visuomotor skill and then learn the skill in that action space. On a higher-level abstraction skill composition, Fang et al.[56] propose a technique that plans multi-step manipulation skill sequence heirarchically using learned latent representations, showing effectiveness to reason about dynamics in long-horizon manipulation tasks. Beyond compositional skill planning, Mandlekar et al.[57] address factorization of task structure from demonstration, while Li et al.[58] use causal discovery to infer structure and interactions that have causal effect on the behavior of the dynamical system.

Increasing capabilities and understanding of robot learning will empower the efforts in enabling highly autonomous systems. To harness this progress for space-borne robotic systems, it is therefore necessary to develop these algorithmic and operational elements of autonomy in a consistent framework. Next section provides details of such a framework.

3. Autonomy Framework Design

Motivated by the lack of autonomy formalism for space systems and inspired by advances in terrestrial robotic autonomy, this section presents a framework for directing space systems development towards generalizable autonomy. This may serve as a template within which space systems in general and robotic OSAM capabilities in particular may be incrementally developed.

Autonomy of a system is the ability to achieve goals while operating independently of external control, using real-time sense-plan-act sequences. Here, this self-sufficiency of the system is decoupled into two elements: *independence* and *adaptability*.

- *Independence* outlines the time horizon and the degree to which a system handles the decision-making, without operator intervention.

- *Adaptability* outlines the degree to which system can adapt to environment variation and the extent it can adapt a task in a goal-oriented manner.

This de-coupling of independence allows a more systematic and granular representation of autonomous system, reducing ambiguity of capability expression.

3.1 Autonomy Levels

Autonomy level scales are important in conveying capabilities of a system in a snapshot without necessitating or requiring knowledge of the functional implementation of the autonomy stack. This is relevant for consistent understanding or representation of autonomous behavior of the system among customers, system engineers and developers of autonomy functions. Similar to taxonomy of autonomous road vehicles (SAE-J3016 [†]), taxonomy standardization with accurate autonomous behaviour descriptions can aid the developing commercial OSAM industry in consistent representation of operational features. Currently, ECSS Space Segment Operability standards[‡] provide a coarse autonomy scale that allows broad categorization of mission execution. The standard lacks granularity beyond operational scheduling and procedures to express generalizability as well as robotic capabilities that are enabled by algorithmic features. On the other hand, autonomy levels in SAE-J3016 levels do not clearly decouple independence and adaptability aspects. We decouple independence and adaptability aspects to clearly express autonomy levels and the corresponding behaviour. These are shown in Tables 1 & 2. Despite generality, the representative example use case is assumed to be robotic manipulation for OSAM, wherever necessary.

3.1.1 Independence

Level I1: Decision making is operator dependent. Time horizon for autonomous operation is usually very small, typically involving time for arm traversal between operator commands. Example: Set-point based teleoperation with state/vision feedback aided by onboard kinematic/inverse-kinematic planner, where autonomy time horizon is the duration of arm traversal between set points.

Level I2: Decision-making is human-dependent and system may additionally handover complete control to the operator when system encounters a potentially unsafe state. Time horizon for autonomous operation is usually small, requiring manual feedback/intervention while executing a sense-plan-act loop. Example: Teleoperation with high-level operator commands like visual servo to a predefined relative arm pose to a visually locked feature target subject to dynamic constraints.

[†] SAE-J3016A: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles

[‡] ECSS-E-ST-70-11C: ECSS Space segment operability (31 July 2008)

Independence			
Level	**Name**	**Decision making**	**Time Horizon**
1	Minimal Autonomy	Operator	Short (vision/motor function execution)
2	Assistive Autonomy	Operator	Short (visuomotor skill execution)
3	Supervised Autonomy	System-Operator	Intermediate (visuomotor skill compositions)
4	Active Autonomy	System	Long (task execution)
5	Full Autonomy	System	Long (tasks/mission execution)

Table 1: Autonomy Levels representing independence of the system

Adaptability			
Level	**Name**	**Environment Adapabtability**	**Task Adapatbility**
1	Rigid Autonomy	None/Minor	None/Minor (Functionally rigid)
2	Robust Autonomy	Minor	Minor (Behaviorally-rigid)
3	Adaptive Autonomy	Partially unknown environment	Intermediate (Cognitive skill composition)
4	Partial Generalizable Autonomy	Familiar environments	High (Task structure adaptability)
4	Full Generalizable Autonomy	Unknown environments	Long (task structure adaptability with online concept generation)

Table 2: Autonomy Levels representing adaptability of the system

Level I3: System executes sense-plan-act loop with operator supervision. Decision making is shared between human operator and the system autonomy stack, requiring decision validation and intervention for critical events. The system may spontaneously hand-over control to the operator on violating confidence bounds, safety margins or safety overrides. Example: Autonomous final approach and robotic capture of an uncooperative spacecraft, with operator validation for critical decisions.

Level I4: System executes sense-plan-act loop without operator intervention, aided by an on-board autonomy stack. Decision making is handled entirely by the system, with operator in an optional and passive monitoring mode. System can compose basic skills to execute fixed and pre-defined tasks and subtasks, over a sufficiently long time horizon. Example: Autonomous vision-based assembly of a telescope sub-structure from semantic task description.

Level I5: System executes sense-plan-act loop without operator intervention, aided by an on-board autonomy stack. Decision making is entirely handled by the system at a task level, once a task description is provided. System can compose task structure to accomplish undefined or softly-defined tasks, while operate autonomously for long time horizons, including significant or entire duration of mission. Example: Autonomous robotic refuelling and re-targeting of an on-orbit fuel station over a life-cycle with dynamic re-fueling assignments.

3.1.2 Adaptability

Level A1: Minor or no environment variability is expected. and operation is done using fixed and pre-defined designed task structures and target objects. Example: Refuelling of a known cooperative spacecraft with target-specific end-effector interface.

Level A2: Minor environment variability is expected within a pre-defined operational environment. The system pre-programmed at a behavioral level and operates using a fixed task/task structure specified in advance. Example: Robotic debris capture system that adapt to misalignments and large uncertainties in target tumbling rate, but only works for a single pre-defined and known target.

Level A3: Partially unknown environment is anticipated during operation to which system must adapt at a skill or task level. The system is pre-programmed to facilitate cognitive functions such basic context reasoning or modifying a pre-defined task structure to accomplish a task. Example: On-orbit robotic inspection and characterization of a dynamically assigned target.

Level A4: The system is able to accomplish tasks in unknown environments^{+} with loose structural similarity to training or test environment. The system is able to reason about a semantic task description or task specifications to synthesize dynamic task structure and skill composition with structural primitives and past experience. Example: Autonomous assembly of large structures for arbitrary sys-

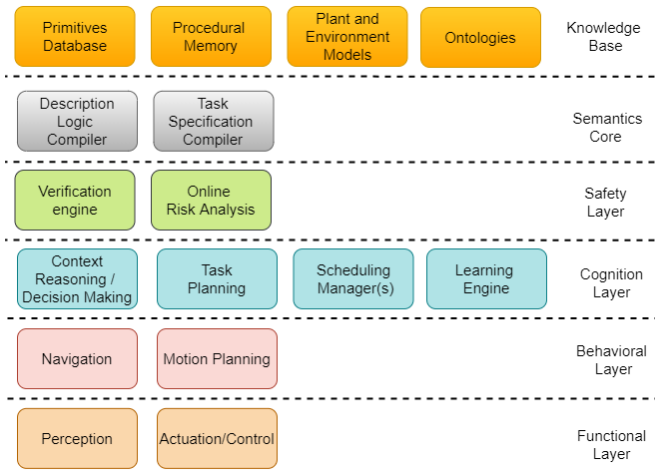

Fig. 1: General autonomy stack abstraction

tems from standard and non-standard sub-components.

Level A5: The system is able to reason about radically new environments[+] by synthesizing new concepts of operation at a task, skill, model or process level. The system can dynamically construct novel task structure and learn composite skills in the new environment, to augment generic primitives and past experience. Example: Autonomous assembly of arbitrary systems with non-standard sub-components in collaborative and non-collaborative environments in presence of humans or other agents.

3.2 Autonomy Stack Abstraction

The autonomy stack that can enable aforementioned levels of autonomy can now be conceived in terms of high-level abstraction layers as shown in Figure 2. The functional layer at the bottom represents elements of perception and actuation, that interface directly with the hardware to process raw sensor data into structure, or process motion-plan into actuator signals. Behavioral layer above represents elements that guide the spatial-temporal behavior in and of the environment like localization, mapping and trajectory optimization. Above this, the cognitive layer allows the system to reason about higher level information, which includes includes scene understanding, dynamic task planning and scheduling, intelligent decision making and on-line learning. The cognitive layer reinforces behavioral layer by processing, encoding and decoding a complex scene information on a high-dimensional representations other than dynamic motion. Given the safety-critical nature of spacecraft operations, the safety layer includes elements like risk quntification, uncertainty representation and behavior verification, to ensure that a goal oriented plan or action does not violate safety primitives, in a system that may be constantly learning and evolving in its environment. The semantics core embeds ability for the system to understand and reason about semantic representation of tasks and operations. Finally the knowledge base consists of predefined structural or functional primitives, dynamic models and ontologies in a declarative manner and a procedural memory that updates and evolves during operation.

3.3 Autonomy Execution Architecture

The autonomy execution architecture develops a template of how autonomous systems can reason about high level semantic task assignments and transform them into real-world execution. While it is meant to describe a flow of execution in a fully autonomous system, the execution architecture is generic and can be used to represent lower autonomy levels, where parts of the flow may be hard-coded, executed on-line and/or executed by an operator. The architecture is depicted in Figure 2 with expansion of subtask execution manager block.

Three capability levels are used: skill (lowest), subtask (intermediate) and task (highest). The execution flow starts with the semantic task description. The system utilizes the pre-built ontologies to first validate the task description and compile logic-based and quantitative task specifications that can be verified in the loop. Subsequently, utilizing the existing knowledge and constraints, the task specifications are semantically validated in terms of whether the task can be accomplished, given the capabilities of the system at the given time. Once validated, the system synthesizes task structure by breaking the top-level task specification into a subtask tree or a graph. Each subtask is then decomposed into elementary skills that are required to accomplish it. Skill assignments are the re-composed into a functional task execution plan with a subtask queue and a skills-schedule per subtask. The functional plan is then verified for initiating execution. Once initiated, the subtask manager handles execution of the subtask queue and making real-time adjustments to schedules and dynamic re-planning of skill execution. This execution schedule is then updated in individual skill execution managers. Skill execution manager takes care of proposing visuomotor actions for a short-time horizon, which are propagated and checked for violation of predefined risk criteria and margins. After the safety of propagated behaviour is verified, the skills are executed. On skill execution, subtask manager is sent an update about the latest state of visuomotor skill execution and the environment. If actions proposed by skill execution manager violate a given risk criteria, the skill execution is bypassed

[‡*] More advanced human-robot interactions including shared goal and/or intentions in a workspace may also be incorporated here

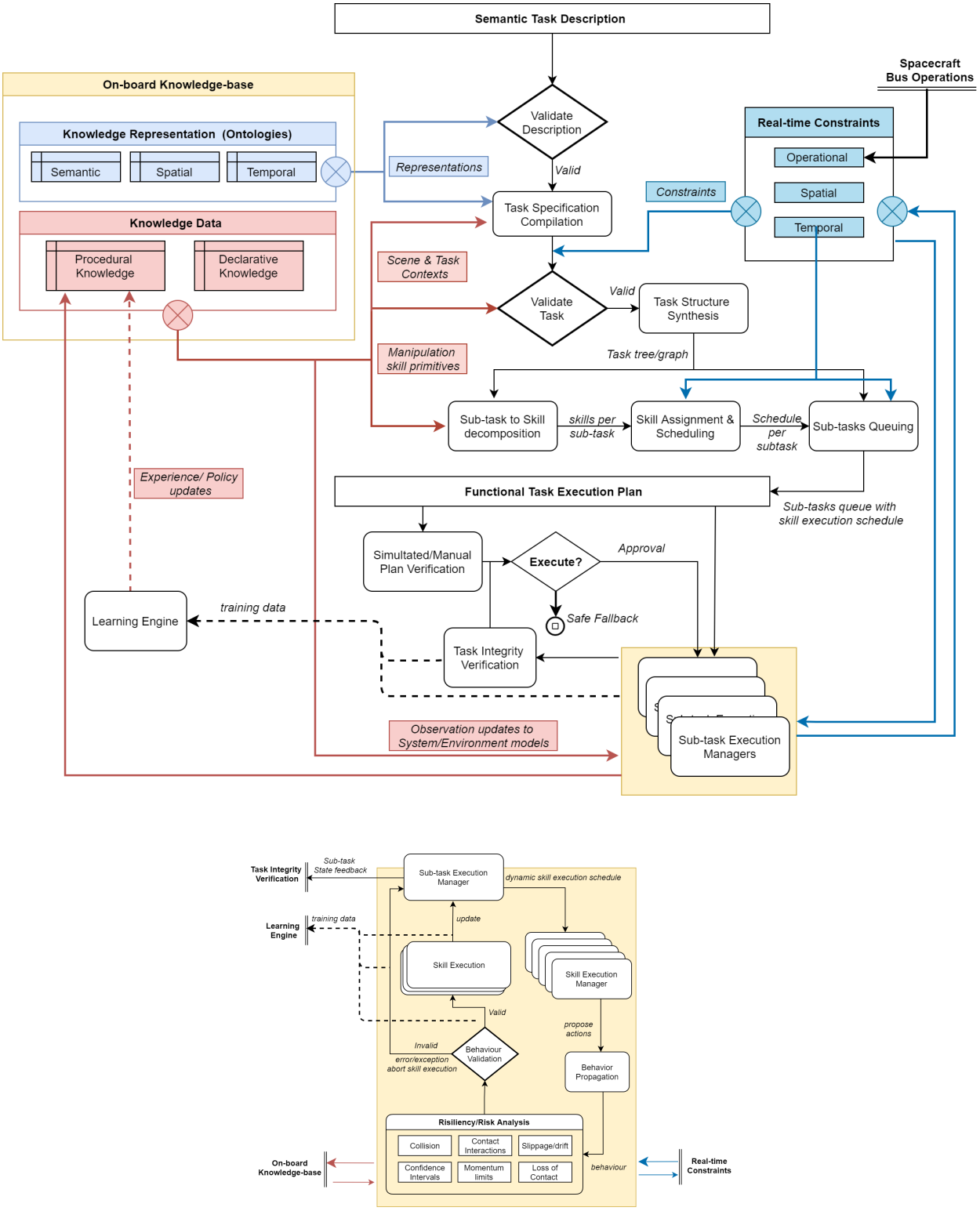

Fig. 2: Generic autonomy execution architecture for enabling goal-oriented autonomous behaviour from semantic long-horizon task description.

and the subtask manager is informed. subtask manager can then choose to dynamically re-schedule and re-assign skills to retry or recover in order to fulfil the state of the subtask. In the process, the subtask manager continuously updates the state of subtasks in a verification block. This block verifies the integrity of the task and the probability of achieving the end-state from the current subtask execution state. The state data of the system, environment, skill execution and subtask execution is sent to a learning engine. The learning engine block is kept generic here to avoid non-trivial specifications. This blcok updates the procedural memory with improved policies and models that enable system to improve with experience.

4. Discussion

The Execution architecture is built with the assumption that the autonomous system will be assigned high-level tasks along its lifetime in a semantic format, which the system would have to reason about and execute successfully. Unlike many of the terrestrial robotic applications, the use of natural language for task description and instructions is not considered.

Three capability levels- skill, subtask and task can be explained as follows. A task is the operator assignment that encodes the goals for the system for a given time horizon. A subtask is a decomposable part of a task i.e. a node in the task structure. The most elementary level capability is a skill. A functionality of any given skill can span one or more of perception, motion-planning and control elements. Examples include known-body pose estimation (vision), compliant dynamic grasping (motor) and visual servoing (visuomotor). Skills are modular and are unique i.e. they cannot be fully composed by other modular skills.

A system might achieve the end-state of the same subtask specification using dynamic combination of visuomotor skills that change according to scene state in real time. The notion of subtask provides an intermediate anchor for representation in between highest level representation (i.e. task) and lowest level representation of goals and states (i.e. skills). It is also assumed that high level tasks and underlying problems are decomposable and re-composable, ignoring the aspect of poor composability where poor factorization of task into subtasks may lead to poor performance on the high-level task on re-composition.

4.1 Example: Autonomous Assembly

Following is a limited breakdown of architectural components in Figure 2 associated with an on-orbit assembly use-case for clarity. Consider an assembly spacecraft equipped with two robotic arms that assembles parts manufactured on orbit similar in concept to Patane et al.[59]

Task: Autonomous structural assembly for wide-field infrared observatory

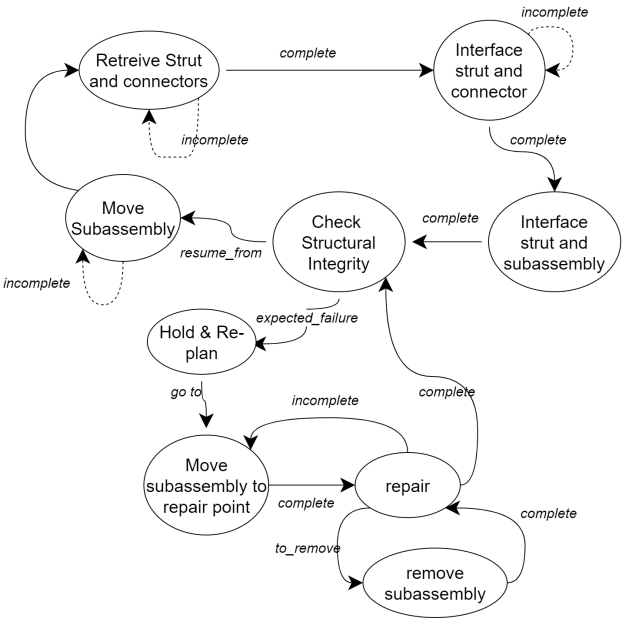

Fig. 3: An example task structure for truss assembly subtask

Task Description: (Semantic form of) Start -> Construct X truss configuration from struts and connectors -> unload and lay mirrors from canister -> unload subsystem modules -> connect subsystem interfaces -> Acquire safe hold position -> Stop

Task Structure: For brevity, consider the first subtask - *Construction of X truss configuration*, where X is an array of truss properties. The sample task structure for this segment could be synthesized as shown in Figure 3. Each node in the graph represents an underlying subtask.

Skill Composition: Each subtask (e.g. retrieve strut node) is realized by composing basic skills (e.g. multi-arm motion planning). Consider the simple subtask of retrieving a strut and a connector before interfacing. Figure 4 shows an example of how this subtask could be realized by composing skills through assignmen and scheduling in a dual-arm system.

Execution Managers and Task Integrity Verification: The execution managers are responsible for real-time execution, progress tracking and dynamic re-planning at the respective (skill or subtask) levels. Task integrity verification tracks a criteria representing task success or the possibility of reaching the task end state. As an example, subtask queue and skill schedules may be encoded in a dynamically expanding behaviour tree form. This allows flexible skill/subtask re-planning. The task integrity verification is then tracking all the paths in the behaviour tree that can lead to the task end-state within constraints.

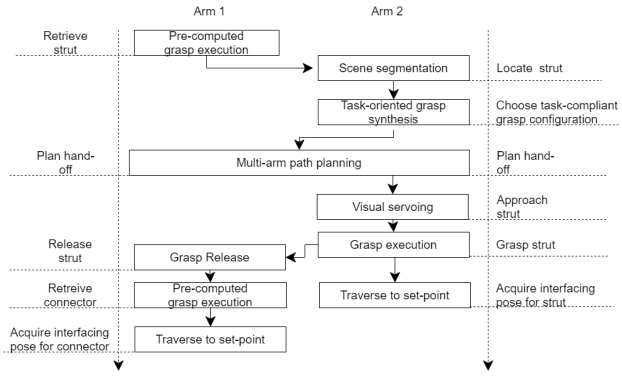

Fig. 4: An example skill decomposition with assignment and scheduling

Learning: The learning engine block represents generic block that allows improvement of system performance based on metrics, success and failure data online. As an example, consider an actuator policy that handles contact-aware execution of secure grasp from a pre-grasp pose. Using a learnt policy eliminates the need for modeling complex contact dynamics for arbitrary materials and geometries. However, a policy learn on ground might be tuned and improved online by tracking pre-defined metrics. Learning engine handles these metrics and software infrastructure for learning and qualification of policy for online updates.

5. Conclusions

This work lays down elements of a software-oriented framework for incremental development of high autonomy in space systems. Targeting the OSAM application domain, the role of vision-based grasping in enabling general manipulation is analyzed using recent progress in data-driven approaches. To enable general manipulation autonomy for space, the need for an incremental autonomy framework is highlighted and elements of the framework are presented. The framework relies on the decomposition of a complex a task into three capability levels- task, subtask and skill. A task describes a long-horizon assignment given to the autonomous system. At the intermediate level, subtasks represent elements of a decomposed task structure. At the lowest level, skills represent modular visuomotor functions that can be dynamically composed to fulfill a subtask. The generic execution architecture uses these capability levels to devise a flow = for translating long-horizon tasks defined by the operator into real-world sense-plan-act routines. The applicability of the framework is demonstrated with an example use case of a highly autonomous system. In the future work, real-time operation, verification and validation elements need to be addressed along with demonstration on a real system.

6. Acknowledgement

This work is supported by the National Research Fund, Luxembourg.

References

[1] R Ambrose, IAD Nesnas, F Chandler, BD Allen, T Fong, L Matthies, and R Mueller. Nasa technology roadmaps: Ta 4: Robotics and autonomous systems. *NASA, Washington DC*, 2015.

[2] Sara A. Carioscia, Benjamin A Corbin, and Bhavya Lal. Ways forward for on-orbit servicing, assembly, and manufacturing (osam) of spacecraft. *IDA Science Technology and Policy Institue Report*, 2018.

[3] Andrew Ogilvie, Justin Allport, Michael Hannah, and John Lymer. Autonomous robotic operations for on-orbit satellite servicing. In *Sensors and Systems for Space Applications II*, volume 6958, page 695809. International Society for Optics and Photonics, 2008.

[4] Andrew Ogilvie, Justin Allport, Michael Hannah, and John Lymer. Autonomous satellite servicing using the orbital express demonstration manipulator system. In *Proc. of the 9th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS'08)*, pages 25–29, 2008.

[5] Jürgen Telaar, Ingo Ahrns, Stéphane Estable, Wolfgang Rackl, Marco De Stefano, Roberto Lampariello, Nuno Santos, Pedro Serra, Marco Canetri, Finn Ankersen, et al. Gnc architecture for the e. deorbit mission. In *7th European Conference for Aeronautics and Space Sciences (EUCASS)*, 2017.

[6] Eve Coste-Maniere and Reid Simmons. Architecture, the backbone of robotic systems. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, pages 67–72. IEEE, 2000.

[7] Rodney Brooks. A robust layered control system for a mobile robot. *IEEE journal on robotics and automation*, 2(1):14–23, 1986.

[8] R Peter Bonasso, David Kortenkamp, David P Miller, and Marc Slack. Experiences with an architecture for intelligent, reactive agents. In *International Workshop on Agent Theories, Architectures, and Languages*, pages 187–202. Springer, 1995.

[9] Richard Volpe, Issa Nesnas, Tara Estlin, Darren Mutz, Richard Petras, and Hari Das. The claraty architecture for robotic autonomy. In *2001 IEEE Aerospace Conference Proceedings (Cat. No. 01TH8542)*, volume 1, pages 1–121. IEEE, 2001.

[10] Guillaume Brat, Ewen Denney, Kimberley Farrell, Dimitra Giannakopoulou, Ari Jónsson, Jeremy Frank, Mark Boddy, Todd Carpenter, Tara Estlin, and Mihail Pivtoraiko. A robust compositional architecture for autonomous systems. In *2006 IEEE Aerospace Conference*, pages 8–pp. IEEE, 2006.

[11] Jeremy Frank, Ari K Jónsson, and Paul Morris. On reformulating planning as dynamic constraint satisfaction. In *International Symposium on Abstraction, Reformulation, and Approximation*, pages 271–280. Springer, 2000.

[12] Vandi Verma, Tara Estlin, Ari Jónsson, Corina Pasareanu, Reid Simmons, and Kam Tso. Plan execution interchange language (plexil) for executable plans and command sequences. In *International symposium on artificial intelligence, robotics and automation in space (iSAIRAS)*, 2005.

[13] Jorge Ocón, Juan Manuel Delfa, Alberto Medina, Daisy Lachat, Robert Marc, Mark Woods, Iain Wallace, Andrew Ian Coles, Amanda Jane Coles, Derek Long, et al. Ergo: A framework for the development of autonomous robots. In *14th Symposium on Advanced Space Technologies in Robotics and Automation*, 2017.

[14] Conor McGann, Frederic Py, K Rajan, H Thomas, R Henthorn, and R McEwen. T-rex: A model-based architecture for auv control. In *3rd Workshop on Planning and Plan Execution for Real-World Systems*, volume 2007, 2007.

[15] Herman Bruyninckx. Open robot control software: the orocos project. In *Proceedings 2001 ICRA. IEEE international conference on robotics and automation (Cat. No. 01CH37164)*, volume 3, pages 2523–2528. IEEE, 2001.

[16] M Muñoz Arancón, Giuseppe Montano, Malte Wirkus, Kilian Hoeflinger, Daniel Silveira, Nikolaos Tsiogkas, Jérôme Hugues, Herman Bruyninckx, Iulia Dragomir, and Ali Muhammad. Esrocos: a robotic operating system for space and terrestrial applications. In *14th Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA 2017)*, page pp. 1, 2017.

[17] Aviv Tamar, Yi Wu, Garrett Thomas, Sergey Levine, and Pieter Abbeel. Value iteration networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, page pp. 2154–2162, 2016.

[18] Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks: Learning generalizable representations for visuomotor control. In *International Conference on Machine Learning*, pages 4732–4741. PMLR, 2018.

[19] Bruce A Aikenhead, Robert G Daniell, and Frederick M Davis. Canadarm and the space shuttle. *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, 1(2):126–132, 1983.

[20] Roel Boumans and Cock Heemskerk. The european robotic arm for the international space station. *Robotics and Autonomous systems*, 23(1-2):17–27, 1998.

[21] Elliott Coleshill, Layi Oshinowo, Richard Rembala, Bardia Bina, Daniel Rey, and Shelley Sindelar. Dextre: Improving maintenance operations on the international space station. *Acta Astronautica*, 64(9-10):869–874, 2009.

[22] Jeannette Bohg, Antonio Morales, Tamim Asfour, and Danica Kragic. Data-driven grasp synthesis—a survey. *IEEE Transactions on Robotics*, 30(2):289–309, 2013.

[23] Anis Sahbani, Sahar El-Khoury, and Philippe Bidaud. An overview of 3d object grasp synthesis algorithms. *Robotics and Autonomous Systems*, 60(3):326–336, 2012.

[24] Jeffrey Mahler, Florian T Pokorny, Brian Hou, Melrose Roderick, Michael Laskey, Mathieu Aubry, Kai Kohlhoff, Torsten Kröger, James Kuffner, and Ken Goldberg. Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1957–1964. IEEE, 2016.

[25] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. Efficient grasping from rgbd images: Learning using a new rectangle representation. In *2011 IEEE International conference on robotics and automation*, pages 3304–3311. IEEE, 2011.

[26] Antonio Morales, Tamim Asfour, Pedram Azad, Steffen Knoop, and Rudiger Dillmann. Integrated grasp planning and visual object localization for a humanoid robot with five-fingered hands. In *2006*

IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 5663–5668. IEEE, 2006.

[27] Carlos Rubert, Beatriz León, Antonio Morales, and Joaquín Sancho-Bru. Characterisation of grasp quality metrics. *Journal of Intelligent & Robotic Systems*, 89(3):319–342, 2018.

[28] Bhubaneswar Mishra. Grasp metrics: Optimality and complexity. In *Algorithmic Foundations of Robotics*, pages 137–166. AK Peters, 1995.

[29] Ravi Balasubramanian, Ling Xu, Peter D Brook, Joshua R Smith, and Yoky Matsuoka. Physical human interactive guidance: Identifying grasping principles from human-planned grasps. *IEEE Transactions on Robotics*, 28(4):899–910, 2012.

[30] Johan Tegin, Staffan Ekvall, Danica Kragic, Jan Wikander, and Boyko Iliev. based learning and control for automatic grasping. *Intelligent Service Robotics*, 2(1):23–30, 2009.

[31] Rosen Diankov. *Automated construction of robotic manipulation programs*. PhD thesis, Carnegie Mellon University, 2010.

[32] Andrew T Miller, Steffen Knoop, Henrik I Christensen, and Peter K Allen. Automatic grasp planning using shape primitives. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 2, pages 1824–1829. IEEE, 2003.

[33] Corey Goldfeder, Peter K Allen, Claire Lackner, and Raphael Pelossof. Grasp planning via decomposition trees. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 4679–4684. IEEE, 2007.

[34] Renaud Detry, Nicolas Pugeault, and Justus H Piater. A probabilistic framework for 3d visual object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(10):1790–1803, 2009.

[35] Lorenzo Pasqualetto Cassinis, Robert Fonod, and Eberhard Gill. Review of the robustness and applicability of monocular pose estimation systems for relative navigation with an uncooperative spacecraft. *Progress in Aerospace Sciences*, 110:100548, 2019.

[36] Sumant Sharma and Simone D'Amico. Neural network-based pose estimation for noncooperative spacecraft rendezvous. *IEEE Transactions on Aerospace and Electronic Systems*, 56(6):4638–4658, 2020.

[37] Tae Ha Park, Sumant Sharma, and Simone D'Amico. Towards robust learning-based pose estimation of noncooperative spacecraft. In *2019 AAS/AIAA Astrodynamics Specialist Conference, Portland, Maine*. AIAA, 2019.

[38] Mate Kisantal, Sumant Sharma, Tae Ha Park, Dario Izzo, Marcus Märtens, and Simone D'Amico. Satellite pose estimation challenge: Dataset, competition design, and results. *IEEE Transactions on Aerospace and Electronic Systems*, 56(5):4083–4098, 2020.

[39] Mihai Andries, Ricardo Omar Chavez-Garcia, Raja Chatila, Alessandro Giusti, and Luca Maria Gambardella. Affordance equivalences in robotics: a formalism. *Frontiers in neurorobotics*, 12:26, 2018.

[40] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. *arXiv preprint arXiv:1703.09312*, 2017.

[41] Sergey Levine, Peter Pastor, Alex Krizhevsky, Julian Ibarz, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37(4-5):421–436, 2018.

[42] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2901–2910, 2019.

[43] Douglas Morrison, Peter Corke, and Jürgen Leitner. Learning robust, real-time, reactive robotic grasping. *The International journal of robotics research*, 39(2-3):183–201, 2020.

[44] Joseph Redmon and Anelia Angelova. Real-time grasp detection using convolutional neural networks. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1316–1322. IEEE, 2015.

[45] Claire Dune, Anthony Remazeilles, Eric Marchand, and Cédric Leroux. Vision-based grasping of unknown objects to improve disabled people autonomy. In *Robotics: Science and Systems Manipulation Workshop: Intelligence in Human Environments.*, 2008.

[46] Gary M Bone, Andrew Lambert, and Mark Edwards. Automated modeling and robotic grasping of unknown three-dimensional objects. In *2008 IEEE International Conference on Robotics and Automation*, pages 292–298. IEEE, 2008.

[47] Dirk Kraft, Nicolas Pugeault, Emre Başeski, MILA POPOVIĆ, Danica Kragić, Sinan Kalkan, Florentin Wörgötter, and Norbert Krüger. Birth of the object: Detection of objectness and extraction of object shape through object–action complexes. *International Journal of Humanoid Robotics*, 5(02):247–265, 2008.

[48] Ulrich Viereck, Andreas Pas, Kate Saenko, and Robert Platt. Learning a visuomotor controller for real world robotic grasping using simulated depth images. In *Conference on Robot Learning*, pages 291–300. PMLR, 2017.

[49] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018.

[50] Shuran Song, Andy Zeng, Johnny Lee, and Thomas Funkhouser. Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations. *IEEE Robotics and Automation Letters*, 5(3):4978–4985, 2020.

[51] Leslie Pack Kaelbling. Foundations of learning in autonomous agents. *Robotics and Autonomous Systems*, 6(2), 1991. Also published in *Toward Learning Robots*, W. Van de Velde, Ed., MIT Press, 1991.

[52] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020.

[53] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015.

[54] Roberto Martín-Martín, Michelle A Lee, Rachel Gardner, Silvio Savarese, Jeannette Bohg, and Animesh Garg. Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1010–1017. IEEE, 2019.

[55] Arthur Allshire, Roberto Martín-Martín, Charles Lin, Shawn Manuel, Silvio Savarese, and Animesh Garg. Laser: Learning a latent action space for efficient reinforcement learning. In *ICRA 2021*, 2021.

[56] Kuan Fang, Yuke Zhu, Animesh Garg, Silvio Savarese, and Li Fei-Fei. Dynamics learning with cascaded variational inference for multi-step manipulation. In *Conference on Robot Learning, 2019*, 2019.

[57] Ajay Mandlekar, Fabio Ramos, Byron Boots, Silvio Savarese, Li Fei-Fei, Animesh Garg, and Dieter Fox. Iris: Implicit reinforcement without interaction at scale for learning control from offline robot manipulation data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4414–4420. IEEE, 2020.

[58] Yunzhu Li, Antonio Torralba, Anima Anandkumar, Dieter Fox, and Animesh Garg. Causal discovery in physical systems from videos. *Advances in Neural Information Processing Systems*, 33, 2020.

[59] Simon Patane, Eric R Joyce, Michael P Snyder, and Paul Shestople. Archinaut: In-space manufacturing and assembly for next-generation space habitats. In *AIAA SPACE and astronautics forum and exposition*, page 5227, 2017.