PILLAR ROBOTS

Purposeful Intrinsically motivated
Lifelong Learning Autonomous Robots

D3.1
First report on development of Smart Attention Module

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement no. 101070381.

1. Change Control

1.1. Document Properties

Name deliverable	**First report on development of Smart Attention Module**
Deliverable No.	**3.1**
Work Package title	**Integrating purposes, motivations and goals**
Author/s	**Retsinas, George; Filntisis, Panagiotis; Oikonomou, Paris; Maragos, Petros, Mattioli, Francesco; Cartoni, Emilio; Cioccolini, Gianluca; Santucci, Vieri Giuliano**
Contributor/s	**-**
Reviewer/	**Duro, Richard**
Editor/s	**Retsinas, George**
Dissemination Level	**Public**

1.2. Revision History

Version	Date	Change	Modified by
0.1	28-9-2023	First Complete Version	G. Retsinas
0.2	07-10-2023	Update Executing Summary & Conclusions	G. Retsinas
0.3	10-10-2023	Add Structure Overview	G. Retsinas
0.4	19-10-2023	Approved by the Consotium with no change	-
1.0	20-10-2023	Final	-

2. Contents

3. Executive summary

This deliverable describes the ongoing research towards a perception-based attention module that will assist the robot's exploration. This work belongs to Work Package 3 - Integrating purposes, motivations and goals of the PILLAR-Robots project and specifically the first task of the Work Package, namely Task 3.1: *Design of the purpose-driven visual attention*. The core idea is to provide the necessary visual attention mechanism that will bias a robot's exploration towards purpose-related regions of the environment.

Specifically, visual attention is investigated under different perspectives, namely effect-driven (pertaining to T3.1.1 *Implementation of the effect-driven Smart Attention Module*) and representation-driven attention (pertaining to T3.1.2 *Implementation of the representation-driven Smart Attention Module*). The former, guides the attention to the robot to regions of the environment that are interesting after a change / action is occurred. This is related to temporal attention and can be interpreted as optical flow or saliency, assuming a low-level vision viewpoint. The latter assumes representations of purpose, typically realized through visual examples.

Moreover, the first version of the Smart Attention Module is developed and describe in detail, combining ideas from the aforementioned visual attention categories. Smart Attention module can be considered as the output of the whole task T3.1, acting as a common framework that includes different sub-modules (e.g., both effect- and representation-driven). It has the operationalization aim of supporting and unlocking the rest tasks of WP3, namely Task 3.2 *Developing purpose-driven autonomous goal discovery/generation mechanisms* and Task 3.3 *Integrating motivational drives within the goal-selection module*, and it will be a key component of the final integration task of WP3, namely Task 3.4 *Integration of the motivational engine and identification/development of other typologies of motivational drives*. In more detail, the Smart Attention Module aims to provide all the necessary functionalities in order assist the robot's focus (e.g., where to go next), following the strategy imposed by the higher-level motivational engine. Key elements of the developed module are: 1) discovering and storing new objects as templates, 2) ignoring robot parts and treating them as attention distractors, and 3) counting interactions with classes of objects. This way, the attention module is versatile under different environments, even with unknown elements, and also versatile under different motivation strategies (e.g., prioritizing discovery vs purpose).

4. List of Abbreviations

Abbreviation	Explanation
WP	Work Package
ARC	Athena Research Center
CNR	Consiglio Nazionale delle Ricerche
UDC	Universidade da Coruña
SU	Sorbonne University

5. List of Figures

6. Introduction

The deliverable describes the work done by ARC for task T3.1 *Design of the purpose-driven visual attention*, in close collaboration with CNR, which leads the WP3, and AI2Life. The basic objective is to provide the necessary *visual attention mechanisms* that will *bias* a robot's exploration towards purpose-related regions of the environment.

Initially, we distinguished two different viewpoints to tackle attention: 1) effect-driven and 2) representation-driven attention. In 1), we solely rely on observing the effects of the robot's exploration and its interaction with the environment (e.g., detect movable objects), while 2) refers to correlating explicit purpose representations with the visual scene. Both approaches have the same end goal, namely facilitating an attention guidance, where the system's attention gradually becomes more focused to environment objects/ elements that correlate with the defined purpose.

In the former category, we can augment the attention of the system by observing possible effects from interaction. For the latter category, we aim to provide the necessary vision tools to analyze each scene and correlate parts of the scene with a purpose-related elements. If the effect-driven approach is not triggered by random exploration, but instead the effect is instigated by a purpose-driven action, we have a more complex cross-dependency of the two categories.

In this deliverable, we will discuss on these different approaches and their capabilities, and we will present the first version of the Smart Attention Module that aims to provide all the necessary functionalities in order assist the robot's focus, regardless the strategy followed by the higher-level motivational drives of T3.2 and T3.2.

6.1. Deliverable Organization

The deliverable is organized around the ongoing subtasks of T3.1, namely T3.1.1 Implementation of the effect-driven Smart Attention Module and T3.1.2 Implementation of the representation-driven Smart Attention Module, as well as the unified framework of the so-called Smart Attention Module that will be the final outcome of the task T3.1. Specifically, the deliverable is structured as follows:

Simulation Environment

This section provides a brief description of the Tiago Simulator, developed by AI2Life and CNR, in collaboration with ARC. This simulator is an enabling technology that provides different simulation environments, able to reproduce functionalities of the Tiago robot, and paves the way towards implementations on realistic environments. The development of such a simulation was of outmost importance in the initial steps of task T3.1 in order to have the necessary visual data for validating the developed algorithms.

Effect-driven Attention

This section describes algorithms that guide attention by observing the effects of the robot's exploration and its interaction with the environment (e.g., detect movable objects). Two main categories of computer vision algorithms were considered: optical flow and saliency.

Representation-driven Attention

This section provides the first version of representation-driven attention that correlates purpose representations, provided as visual examples, with the visual scene. For this version, purpose is considered as a visual example of a single object (e.g., apple). We used two state-of-the-art approaches, one for scene segmentation and one for vision-text common encoding, under a common framework to quickly adapt under different visual domains.

Smart Attention Module

Here, we describe the developed Smart Attention framework that includes different sub-modules (e.g., both effect- and representation-driven). In practice, we focused on the representation-driven idea, following the success of the preliminary results in the simulated environments. This framework provides all the necessary functionalities in order assist the robot's focus (e.g., where to go next), following the strategy imposed by the higher-level motivational drives. Key elements of the developed module are: 1) discovering and storing new objects as templates, 2) ignoring robot parts and treating them as attention distractors, and 3) counting interactions with classes of objects.

6.2. Connection to Tasks/WPs

The reported task (T3.1 *Design of the purpose-driven visual attention)*, as described previously, is a part of the motivational engine (T3.4) and thus affects and is affected by motivational engine's sub-component that realize tasks T3.2 and T3.3. These dependencies are visualized in Figure 1. The smart attention module is implemented via the two different perspectives on attention (effect- vs representation-driven attention). Moreover, this task is very correlated with the other perception tasks of ARC (namely T2.2, T2.3 and T2.4) since perception / visual modules of WP2 are the building blocks of the described attention mechanisms and in the future more developed modules of WP2 can be integrated in T3.1. Finally, T4.2 Generating Visual Representations from Attention of WP4, that starts after Y1, is a closely correlated task conceptually and thus results/outcomes of T3.1 can be beneficial to T4.2.

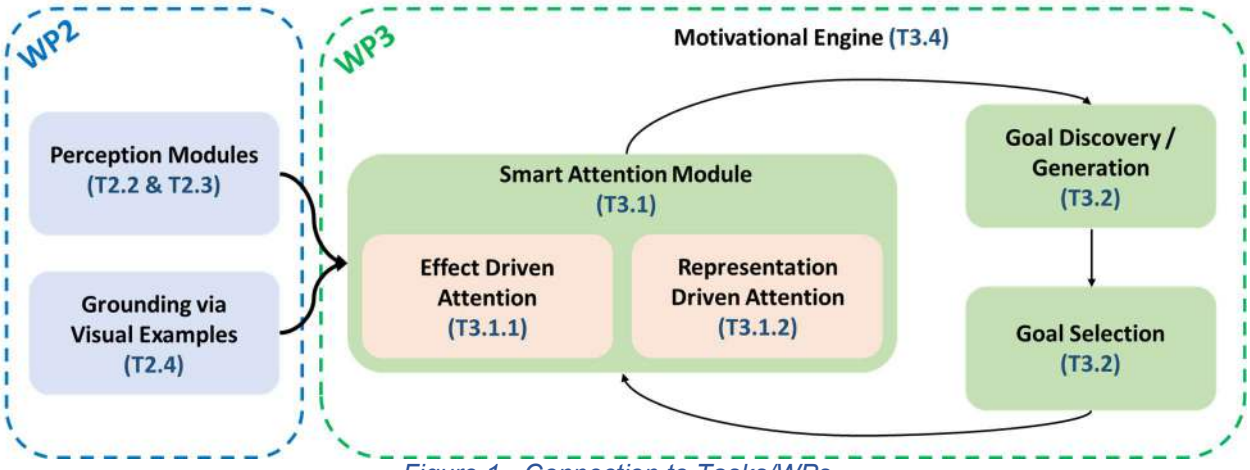

Figure 1 - Connection to Tasks/WPs

7. Simulation Environment

Exploring the concept of (purpose-driven) attention is limited by the existing visual data, highlighting the necessity of simulated environments. Ideally, these simulation environments should include the Tiago robot, to be as close as possible to the actual application scenarios to be tested in PILLAR. Therefore, a TIAGO simulator was developed by AI2Life/ CNR, in collaboration with ARC. This simulator includes a set of simulated environments. Examples of these environments are: a single table with multiple objects, or a multi-table case, where there are also colored buttons that control the corresponding colored boxes, as seen in the following pictures. The latter scenario will help us to investigate more complex interactions with multi-step dependencies (e.g., open a box to put an object in it).

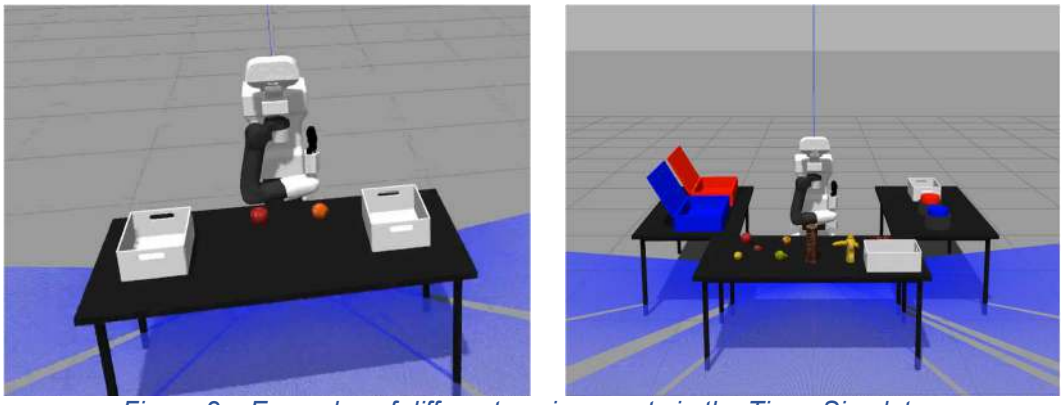

Figure 2 – Examples of different environments in the Tiago Simulator.

Figure 3 illustrates the system architecture responsible for ensuring the operation of the robot and its corresponding virtual environment. This architecture is implemented using Python programming language and interfaces with the underlying ROS (Robot Operating System) system, specifically the noetic version. While the fundamental structure of the overall architecture is well-established, it is important to note that the system is still undergoing development, with new functions being periodically incorporated.

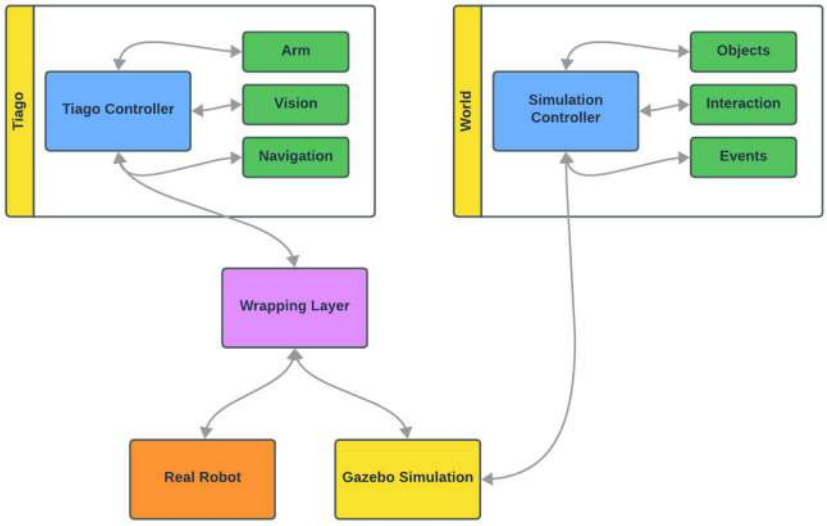

Figure 3 – Tiago Simulator Architecture

The Tiago controller assumes the responsibility of managing the robot's actuators and sensors through three distinct sub-modules:

1. **Vision Module:** This module is responsible for collecting and processing raw visual data obtained from the robot. Additionally, it controls the movement of Tiago's head.
2. **Arm Module:** The Arm Module primarily focuses on the planning and control of the arm and torso movements of the Tiago robot.
3. **Navigation Module** (Currently under development): This module is in the process of being developed and will integrate utility functions for the movement and planning of Tiago, especially through the base equipped with a differential drive motor.

Through a wrapping layer, we have the capability to execute this Python wrapper with both the physical robot and the simulated robot. The simulated world component comprises ROS and Gazebo wrappers written in Python to facilitate interactions with the virtual world.

The controller itself encompasses three sub-modules:

1. **Objects Module:** This module is responsible for controlling the position and physical model information of objects within the simulation.
2. **Interactions Module:** The Interactions Module handles the management of interactions between various objects within the simulation. For instance, it addresses scenarios where excessive force may have been applied to an object, such as an apple.
3. **Events Module:** In an interactive environment, certain scenarios involve dynamic changes. The Events Module is designed to handle such situations. For example, it manages instances where the robot interacts with buttons, subsequently modifying the environment accordingly. This module serves as the crucial link between the simulation in Gazebo and the developer, ensuring seamless interaction and feedback.

8. Effect-Driven Attention

An important aspect of the exploratory phase of the robot is the ability to understand changes in its environment, and the exact location of these changes. In other words, the robot should focus their ***attention*** on these changes, as they may be crucial for discovering properties of the environment. Let's consider the following examples:

1) A robotic arm is randomly moving until it touches an object (e.g., a ball). Then this object starts to move. This movement is the effect of the robotic exploration and can be captured and used to detect the object. It is important to infer that this is a movable object, for further exploration with potentially more complex interaction (e.g., try to grasp it).
2) A button is pushed and a box opens automatically. Here we have two actions, that are independent visually, in the sense that the box may be far from the button. In the ideal scenario, this correlation should be captured and stored, for further exploitation.

We can perceive this task as the collection of fundamental, task-agnostic, vision modules that aim to guide the attention of the robot to "interesting" regions, especially after a change was reported. We distinguish two primary vision modules for this task:

- Optical flow: estimate changes between frames, which indicate object motion.
- Saliency: highlights the region on which people's eyes focus first.

Optical flow

Optical flow in computer vision refers to the pattern of apparent motion or displacement of objects and features in an image or video sequence. Optical flow estimation aims to determine the velocity (speed and direction) of objects and features in an image. It is a critical technique for understanding motion in videos and tracking objects. Several variants of optical flow have been developed, including dense optical flow (computing flow for every pixel) and sparse optical flow (estimating flow only for specific points or features). In our case, we have experimented with two very recent deep learning variants, since classical computer vision approaches have notable limitations (e.g., assuming a small displacement between frames).

Specifically, we used FlowNet2[1] and Perceiver IO[2]. The former is more lightweight (CNN architecture) but less accurate than the latter (Transformer-based architecture). The optical flow frameworks attained the underlying motion even for large displacements, as the following figure highlights. As we can see, the action of dropping a ball is accurately detected to the point that the ball can be segmented correctly. If such a segmentation occurs, then a tracking algorithm can be used for further interacting with the same object.

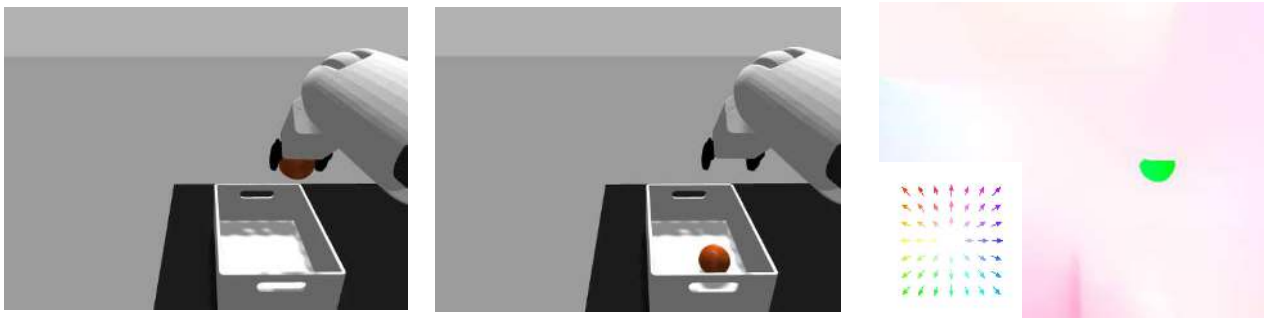

Figure 4 – Optical flow example between two frames. The object is separated highlighting a sudden downward movement (green color).

Note: When the camera is moving, it can change the resulting flow, but in a structured manner. This issue can be addressed, if necessary, but it has not been explored in this work.

Saliency

Saliency in computer vision refers to the identification and highlighting of the most visually distinctive or significant regions or objects within an image or scene. It is a fundamental concept used to understand where human attention is likely to be drawn when viewing an image or video. Saliency analysis generates saliency maps, which are grayscale images that assign saliency scores to pixels or regions within an image. Higher scores indicate more salient or attention-grabbing areas. Following the effect-driven rationale, we are interested in spatiotemporal saliency in videos, which is typically governed by actions.

Again, a deep learning solution is employed, trained in large datasets for video saliency. Specifically,

[1] Ilg, Eddy, et al. "Flownet 2.0: Evolution of optical flow estimation with deep networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.
[2] Jaegle, Andrew, et al. "Perceiver IO: A General Architecture for Structured Inputs & Outputs." International Conference on Learning Representations. 2021.

we employ STAViS[3], a model developed by our team, for audiovisual saliency detection. This method employs a single network that combines visual saliency and auditory features and learns to appropriately localize sound sources and to fuse the two saliencies in order to obtain a final saliency map. The network has been designed, trained end-to-end, and evaluated on six different databases that contain audiovisual eye-tracking data of a large variety of videos. As described, STAViS supports also an audio modality, for a more complete and realistic pipeline, when sounds are also crucial in saliency. This option is not explored yet, since the simulation environment does not include sound. The results of audiovisual saliency for videos of people talking to each other can be seen in Figure 5. This could be a very important feature for the Edutainment scenario.

Following this approach, as we can see in Figure 6, specific regions of the frames are highlighted: the hand of the robot and the objects on the table. It is worth noting that in the first image, the highlighted object is not moving, but is generally a salient object that the human eye tends to focus.

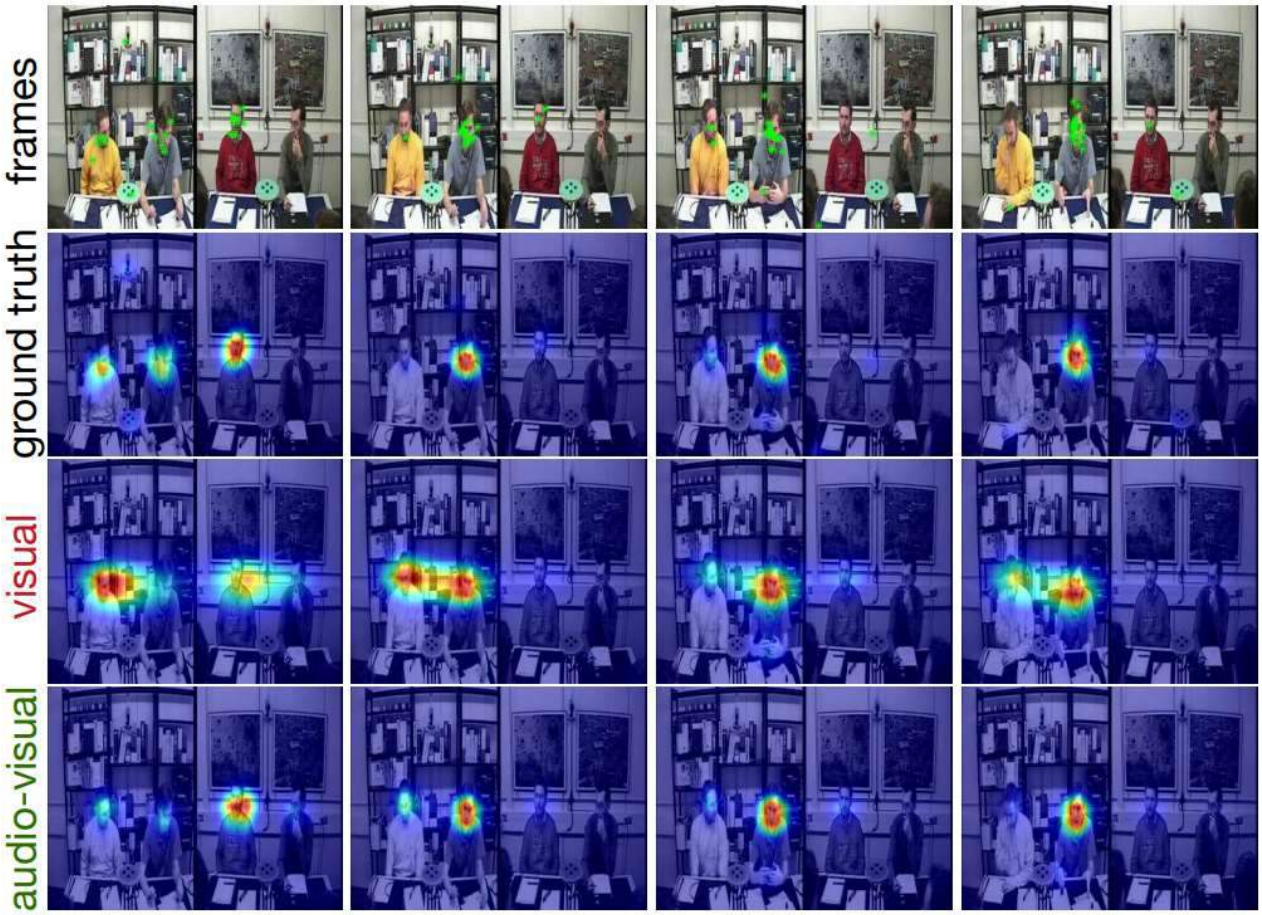

Figure 5 – Saliency results of STAViS.

[3] Tsiami, Antigoni, Petros Koutras, and Petros Maragos. "Stavis: Spatio-temporal audiovisual saliency network." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020.

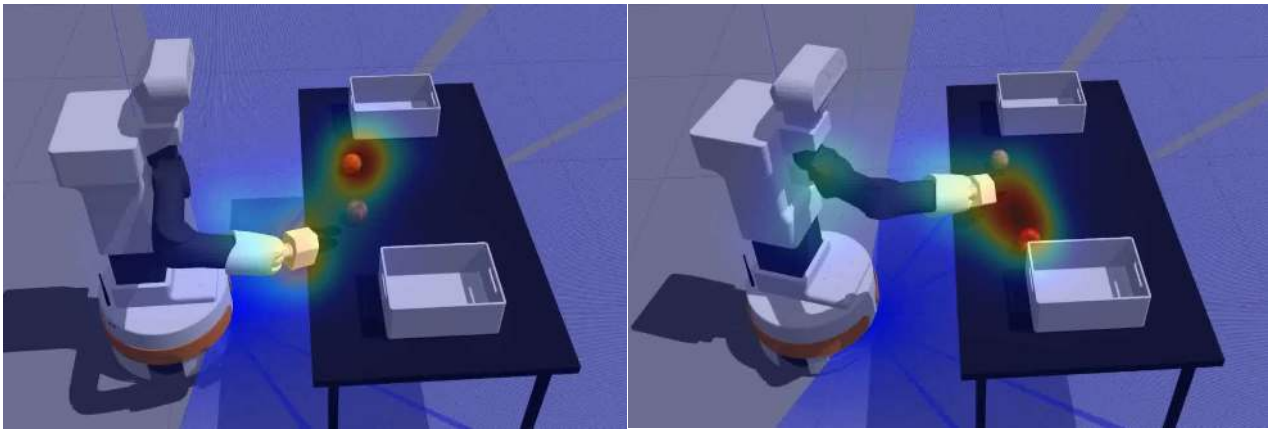

Figure 6 – Visual saliency for a video generated by the Tiago simulator.

Outtakes

Both these approaches, provide a coarse estimation of interesting regions in the image and can further augment an attention system, but they can comprise an attention module on their own, at least in the context of the project and the requested functionality. There are two main problems:

- Many objects / elements of the scene are not recognized and should be contacted with (and create an effect) to be relevant. Ideally, we want the attention model to also guide the exploration. Moreover, the purpose component cannot be integrated into such a pipeline.
- The robotic arm, and the robot in general, is normally moving and thus "attracts" attention. However, our goal is to focus on the scene elements rather than the robot itself.

Implementation-wise, these models are computationally demanding (large Deep Neural Networks), that may not be ideal for approximating a real-time module on a practical application. Nonetheless, if these approaches are deemed necessary, they can be compressed following typical deep learning practices (e.g., pruning, distillation, etc.).

It's worth noting that similar ideas and algorithms are considered for the purpose grounding task of WP2, where we will try to "distill" purpose-relevant information from videos.

9. Representation-Driven Attention

Here, we aim to build a framework that allows us to integrate purpose, in the sense of correlating objects with purpose. The first step is to provide an analytical description of the visual scene and of its elements. Initially, the goal was to use vision modules of WP2 for this task. However, we faced two significant challenges:

1) The existing modules of WP2 are trained on available datasets that may not include elements of interest (e.g., the object detection pipeline may not have a specific object). In such cases, it is very difficult to augment the existing dataset with the requested object. As a sidenote, the capabilities of the modern Generative AI, can provide an effortless solution to this problem: create synthetic images of the requested object. Nonetheless, for each new object, we have to manually account for it, create a set of images and then re-train a new augmented model.
2) Even if all objects are already known, the environment may be vastly different, leading to the underperformance of the object detection module. This was a major issue for the videos

captured through the simulation pipeline; even though object detection works well in the wild, it had under par performance in the more simplistic simulated scenes.

Both these issues relate to the domain adaptation task, that we will further explore in Task 2.2.3.

Segment Anything Backbone

To circumvent the domain adaptation issue, we decided to use a very powerful and recent visual processing tool that performs image segmentation. Specifically, we used the Segment Anything Model (SAM)[4] which segments an image into regions without assuming specific objects or classes. The initial model SAM model is trained on a vast amount of data and thus able to generalize well under very different domains.

It should be noted that Segment Anything can also be used with query inputs: segment a region around a query point or regions that correspond to NLP embeddings. This option can lead to useful future extensions of the presented pipeline.

Implementation-wise, the initial SAM model is rather computationally demanding. To this end we used a variant called FastSAM[5], which uses a YOLOv8 segmentation backbone, as implemented in https://docs.ultralytics.com/models/fast-sam/. Its architecture is depicted in Figure 7, while its advantages can be summarized as:

1) Generic segmentation: No assumption on the set of objects / elements is required.
2) FastSAM is lightweight for close to real-time implementations.
3) Tracking is also supported.

More details on the module itself can be found in Deliverable 2.2.

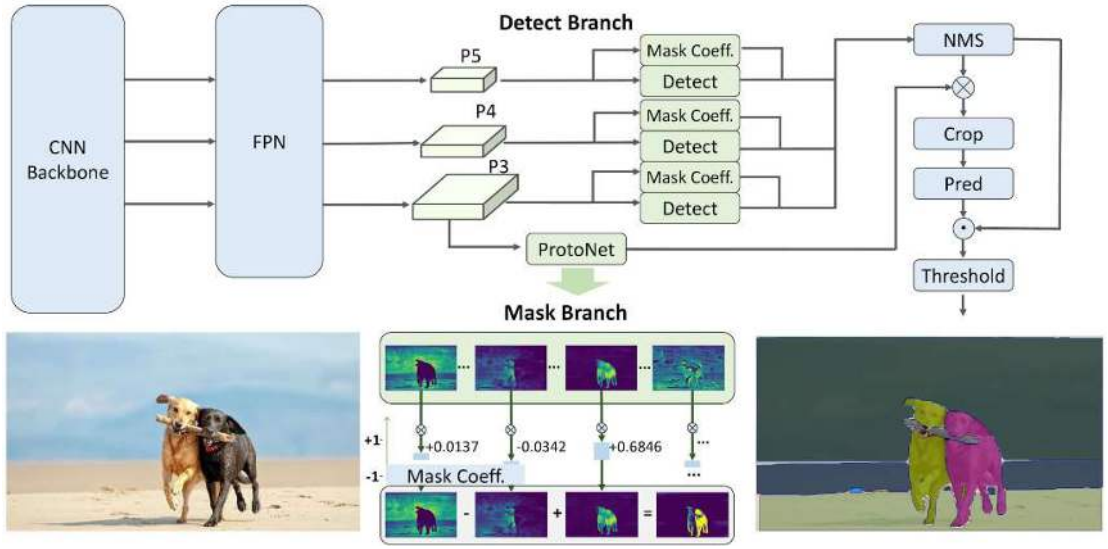

Figure 7 – FastSAM architecture. From https://docs.ultralytics.com/models/fast-sam/.

CLIP (Visual) Features:

[4] Kirillov, Alexander, et al. "Segment anything." arXiv preprint arXiv:2304.02643 (2023).
[5] Zhao, Xu, et al. "Fast Segment Anything." arXiv preprint arXiv:2306.12156 (2023).

Next, we want to correlate segmented regions with purpose indicators. In this context, purpose indicators can be a reference image or a text embedding. Therefore, we considered a common embedding space of visual and textual information. Specifically, we employed CLIP[6] for this task.

CLIP (Contrastive Language-Image Pretraining) is a deep learning model by OpenAI that connects and understands images and text, enabling tasks like image classification and text-to-image generation in multiple languages.

Essentially, CLIP project images and their text descriptions (as taken from the Internet) into a common space that enables cross-modality functionalities. This common projection is trained via contrastive learning. Here's how CLIP applies contrastive learning: CLIP pairs images with related text descriptions as positive examples and pairs them with unrelated text descriptions as negative examples. Then, CLIP minimizes a loss function that encourages positive pairs (matching images and text) to be closer together in the embedding space and negative pairs (mismatched images and text) to be farther apart. The contrastive loss function aims to maximize the similarity between positives and minimize it between negatives. This process is visualized in Figure 8.

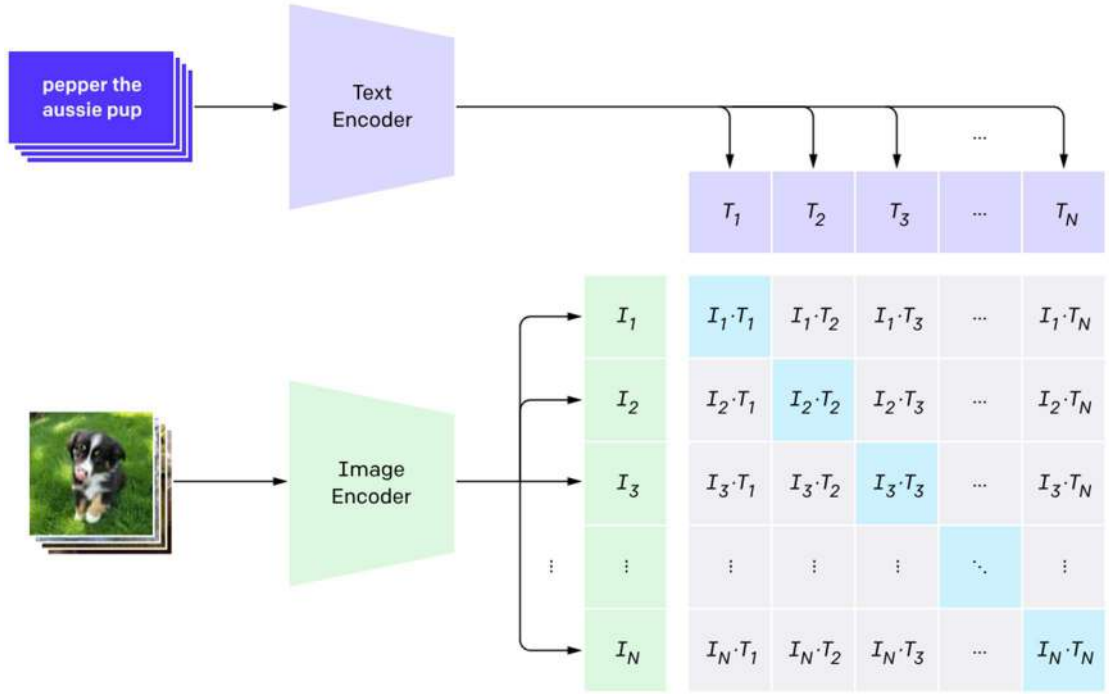

Figure 8 – Overview of the CLIP's common encoding of visual and text modalities. From https://openai.com/research/clip.

At this point, we only use the visual encoder to project objects of the same class from different viewpoints and with different appearance into a common embedding. In other words, we exploit the power of the CLIP model, that is trained with a vast amount of data in order to provide invariant visual features under different transformations / deformations, enhancing the generalization of the system.

[6] Radford, Alec, et al. "Learning transferable visual models from natural language supervision." International conference on machine learning. PMLR, 2021.

An indicative example of a very simple purpose-guiding approach is when specific objects of interest are “grounded” either through a set of template images (e.g., images of a banana from different viewpoints) or from an actual text description, such as “a picture of a banana”. We focus on the former scenario, where the purpose is instantiated through a set of visual examples. The latter, even though supported by the proposed pipeline up to a level of simple textual descriptions, is not yet explored and will be re-investigated along with the progress of the NLP task of WP2.

Then, the purpose-oriented attention can be formulated in a very straightforward way: for each segmented region, provided by the segmentation pipeline of FastSAM, we compare the visual encoding of the region against the encodings of a set of predefined templates. The level of similarity is an attention indicator, expressed as a cosine similarity metric. In other words, we correlate each segment with purpose-related objects, given as grounding templates.

Outtakes

A basic purpose-driven attention system was developed based on an object-agnostic image segmentation algorithm (Segment Anything), a backbone that can break each scene into its elements under various environments without the need to fine-tune. Then each segment can be correlated with purpose indicators, in the form of template images, with the CLIP model. We only used the visual extractor of CLIP to compute the similarity between extracted regions and template, but CLIP also includes the image-to-text similarity capability, that will be explored in the future.

The only limitation of such an approach is the assumption of a purpose template that includes a single object (e.g., a fruit). This is closely related to the purpose grounding task of WP2 and further experimentation with more complex examples of purposes realizations will be explored in the next year.

10. Smart Attention Module

Up to this point, we presented different attention modules from different perspectives, namely effect-driven and representation-driven. In a later stage effect can also be correlated with purpose, if the action taken, aims to achieve a goal. Nevertheless, the simple simulations explored at this point do not contain such complex long-term interactions. This concept should be revisited, if deemed appropriate, in the future.

From now on, a single attention module will be considered, referred to as ***Smart Attention Module***, containing the different described approaches as sub-modules. Our effort is shifted into a seamless integration of all useful functionalities, rather than focusing on the causality of attention mechanisms.

In practice, both the presented effect-driven attention approaches provided “noisy” results, though useful. Their overall functionality is suppressed by the more robust representation-driven approach, at least for the simulation environment, where our algorithms were tested. This is in line with the domain adaptation issue (the used models were trained on data of different nature). Especially when considering their computational complexity, it is safe to consider them as auxiliary components that can be used to further assist the system, but the core pipeline will rely on the segmentation model.

The developed smart attention module has several novel elements that promote robustness:

- Stores unknown objects and finds if the robot interacts with them again.
- Detect robot parts (as known predefined templates) and remove them from possible attention elements.
- Counts the robot interactions with classes of objects in order to assist the exploration phase (with the help of motivational engine) – we may are interested in discovering unknown objects of interact many times with an already known to fully understands its attributes.

Distinguishing known from unknown regions

Our goal here is to consider all the necessary functionalities in order for the smart attention module to assist the robot exploration, following an open-ended learning concept. Thus, we will focus on the very crucial task of distinguish segmented regions into two regions: known and unknown.

For simplicity, we consider the simulation scenario of a robotic gripper interacting with a set of objects on a table. A subset of these objects is known beforehand.

Following the proposed approach of the previous section, we can build a small template "dataset" of all the known objects. Example of these templates are shown below:

Figure 9 – Templates of various simulation objects.

Then, we analyze the recorded scene:

- Segment the scene into regions (with tracking for video sequences)
- Extract the visual embedding of each region using CLIP.
- Compare the extracted visual embeddings to the template dataset.
- If the similarity is above a predefined threshold, the object is considered as known. Otherwise, is considered as unknown.

The aforementioned procedure is similar to the representation-driven pipeline, but there are two differences: 1) the templates do not necessarily correspond to a purpose-related object – we simply try to explore and understand the whole scene, and 2) the known/unknown element is added.

Purpose can be then correlated with these templates even in a later stage, as a meta step, possibly refined by human guidance.

Updating the Template Dataset:

To assist the exploratory phase, we provide the functionality of dynamically updating the set of known objects. In more detail, for an unknown region / object:

- Store this region as a new template and assign to it a unique id. If a new similar object, of the same class, is seen at a later step, we will have its information already stored and we can detect it successfully.
- If the robot selects to interact with the object, we can store the same object from different viewpoints (or even under pressure – e.g., deformable objects) using the same class id, augmenting our already knowledge of its appearance. This step has an underlying effect-driven rationale.

Note that this set of newly "discovered" objects will not correspond to a specific human interpretation (e.g., "fruit"), as done in the set of predefined objects. Nevertheless, in the long run and through the exploration phase, the robot can build useful representations of these new objects and how they interact with their environment.

Ignoring Distractors

Apart from exploring unknown objects, some strategies may include revisiting already known objects (e.g., to further understand the object and its dynamics with the environment, or checking long-term actions and effects). However, specific objects may be negative examples/templates in the context of our purpose-motivated exploration. Their positive /negative impact, that will eventually guide the attention, will be controlled by the purpose grounding step, as well as the overall motivational engine. This concept will be explored in more detail in the future. However, there is a fixed "hard" negative template to take into account; that is the robot itself.

Specifically, we know beforehand, the robot's "anatomy" and we can store visual templates (and their embeddings) of the robot's arm (or other parts that may be in its visual receptive field), as seen in Figure 10. This way, we can detect these parts and treat them as distractors, avoiding assigning any attention to them. This is a very important step to simplify the captured scene and direct the attention to actually useful elements of the scene. A case of ignoring robot parts and boxes is depicted in Figure 11.

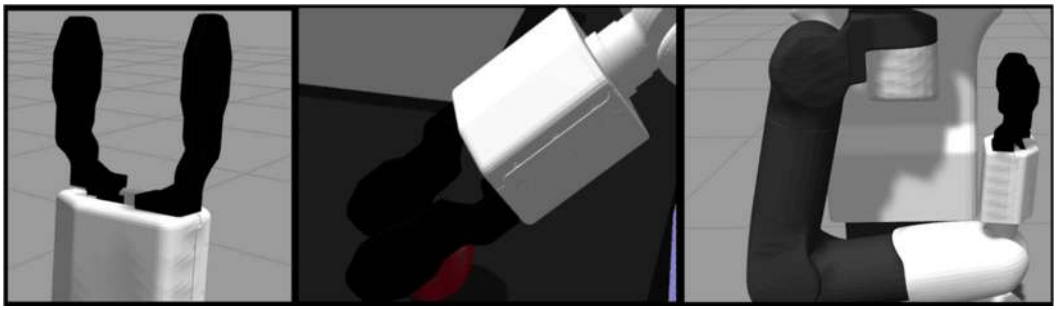

Figure 10 – Templates of robot parts acting as distractors.

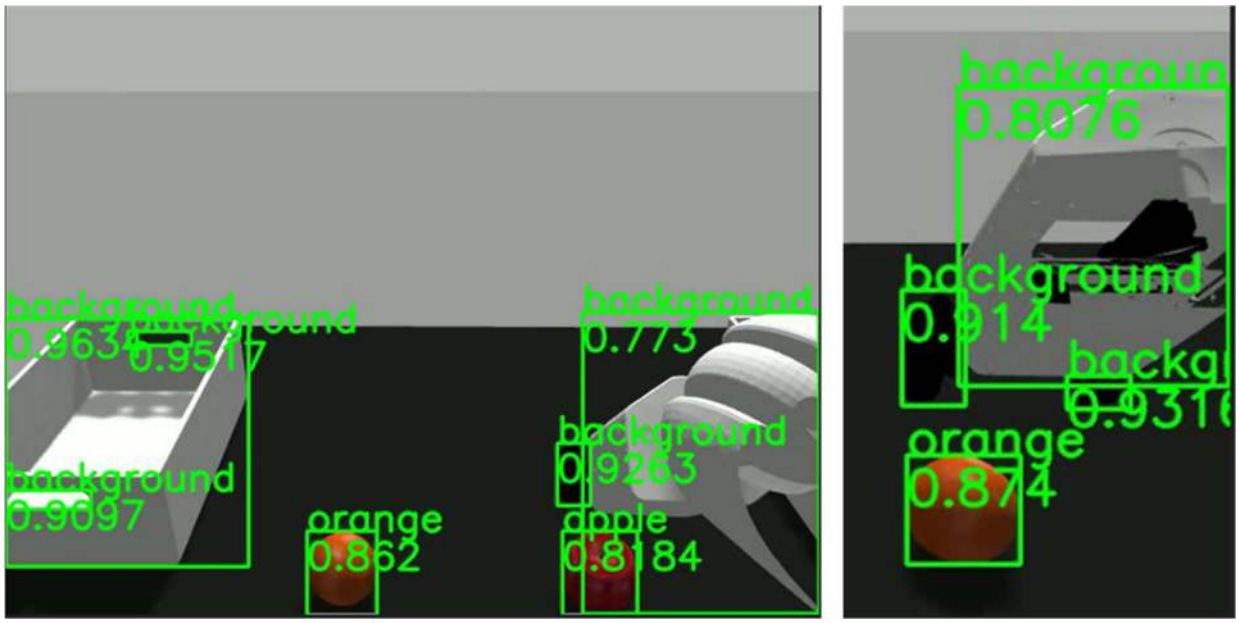

Figure 11 – Examples of the pipeline that tries to separate "interesting" objects from elements treated as background.

Including Purposes

As noted earlier, the presented pipeline does not explicitly request a purpose indicator. Typically, the robot will prioritize exploration/discovery in the first steps, before starting to be more purpose-oriented in a later stage. Therefore, in our framework we consider purpose as external indicators that may added in any time and can be correlated with the existing template dataset. For simplicity, we consider purpose indicators to be given as a different set of object templates (different from our extracted templates). For example, if our goal is to pick apples from the table and place them in a box, we can start by exploring all detected objects. Then, at some point, an external set of images of apples can be given, as a purpose indicator. We should check the similarity of purpose templates with the developed template dataset and assign an extra label to existing templates that are correlated with the provided purpose. This way, we can bias the attention towards purpose-related objects, according to the feedback from the motivational engine.

Measuring our knowledge of an object

The described pipeline provides similarity scores for every detected region with respect to the existing templates. The executed action of the robot is actually controlled by the motivational engine, which provides the necessary biases to select what region should be visited next. Therefore, whether a curiosity-driven (e.g., interact with unknown objects) or a purpose-driven (e.g., interact with purpose-related objects) exploration will be adopted, is a meta-decision, stemming from the motivational engine, rather than the smart attention module. Nonetheless, the smart attention module should be able to facilitate each and every such decision.

For example, one possible strategy is to discover first any "unknown" elements, or to further interact with already known that you have not interacted much, or even interact only with an already known subset of relevant objects towards the completion of a goal.

According to this rationale, we can distinguish an extra important functionality: counting the

interaction with specific object classes (or even specific unique objects, using the tracking capabilities). In more detail, each object class will also contain a counter that represents the reported interactions with this class. In other words, we measure our knowledge of each class of similar objects: if I have interacted with a specific class a lot, we consider that we have a “decent” knowledge of the class. In the long-term, such knowledge of classes could lead to provide more robot-oriented representations of the classes, discovering useful properties / attributes of them that will help specific actions (e.g., grasping). Counting will be implemented as the addition of a scalar value that corresponds to the reported similarity of the object, if this object has such value above the predefined threshold to be considered as a known object. Therefore, we can quantify how much we interacted with similar objects and use it at a higher cognitive level to make the appropriate decisions.

Outtakes

The overall pipeline described in this section is depicted in Figure 12. In a nutshell, we provided all the necessary functionalities in order to build a meta-algorithm that controls exploration and can guide the robot towards purpose related elements. The final out of the system, given a selection strategy, will be a weight per region, where higher weight means higher probability to explore this region in the next step. Formulation-wise, in order to treat region weights as probabilities, we assume that the summation of the region weights in a scene equals to 1. The probabilistic selection of region has an important extra functionality: it allows the robot to visit regions with low probability, since they may seem irrelevant at the current step, that may ``unlock” a multi-step dependency (e.g., push a button that opens a box, where we can later put objects in).

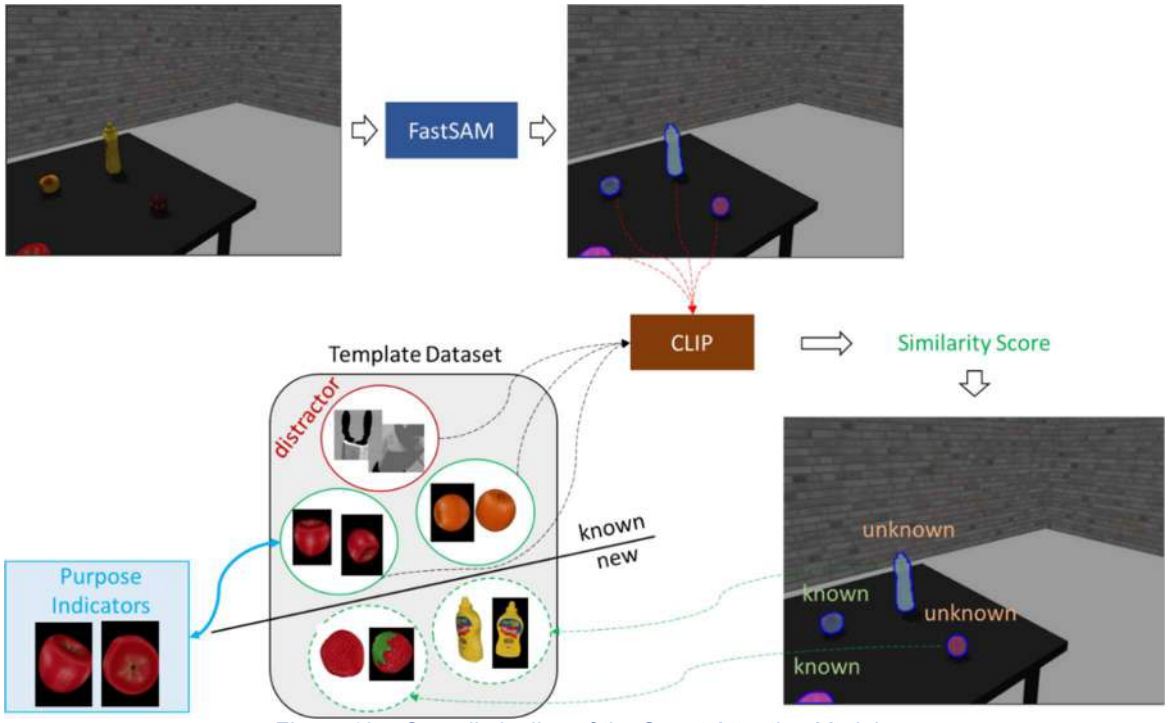

Figure 12 – Overall pipeline of the Smart Attention Module.

11. Conclusions & Next Steps

In this deliverable, we presented the developed Smart Attention module, pertaining to T3.1, along with potentially useful auxiliary sub-modules (e.g., effect-driven attention of T3.1.1 or representation

driven attention of T3.1.2), that aims to assist the robotic exploration, either guided by purpose or not. The underlying concept is straightforward: detect attention "attractors". In the discovery phase, which is linked to intrinsic motivations, such attention regions may be new, unseen, objects that we want the robot to interact with. Towards a more purpose-oriented exploration, which is linked to extrinsic motivations, we want to interact more with objects related to purposes. Following the rationale of visual grounding of purposes through examples (T2.4), we can select objects similar to the purpose templates as attention regions. Overall, the Smart Attention module can be seen as part of the Motivational Engine (T3.4), that affects and is affected by goal discovery (T3.2) and selection (T3.3), acting as a purpose-biased perception system, potentially including several perception modules from WP2.

In more detail, relying on the Tiago simulator, we experimented with several scenarios of robot-object interactions. For these simulations, effect-driven approaches were helpful, but they were underperforming compared to representation-driven approaches. To this end, the effect-driven approaches were considered as auxiliary sub-modules and the representation-driven approach to be the core of the Smart Attention Module. To further enhance the operationalization of the developed Smart Attention module, we developed several useful functionalities, including storing templates of new objects to be recognized in the future, ignoring robot regions and counting interactions with the same class of objects. Thus, we can "analyze" each scene, distinguish known and unknown objects, and measure the level of "knowing" the objects by counting interactions. Such analysis can be straightforwardly used by the goal selection mechanism of T3.3 and the motivational engine in general (T3.4) to guide the robot towards regions of interest according to its motivational drives (e.g., discover things or interact with a purpose-related object).

Even though this first version includes several useful functionalities, there are several extensions to be considered:

- First and foremost, we must combine the developed Attention module with the Motivational Engine of WP3, in collaboration with CNR, to check its overall effectiveness under different exploration strategies.
- Consider complex scenarios that are in line with the project's application, along with the partners, to validate its usefulness, as well as detecting potential missing features.
- Add the NLP-aspect, in collaboration with SU, that can correlate templates with textual description and further correlate them with text-formatted purpose.

As an optional side-goal, we are interested in exploring attention from another viewpoint – as another auxiliary submodule. Specifically, along the lines of NLP, image captioning and image generation, Large Language Models (LLMs) and Transformer architectures have prevailed. Transformers have an innate formulation of attention that can be very useful to our cause, and is already utilized in image editing by guiding visual attention to specific text embeddings. An example is show below. This idea of attention from text can be a powerful tool to be integrated in the Smart Attention module. As it is now, the only downside is the computational complexity of such operation and should be further studied.

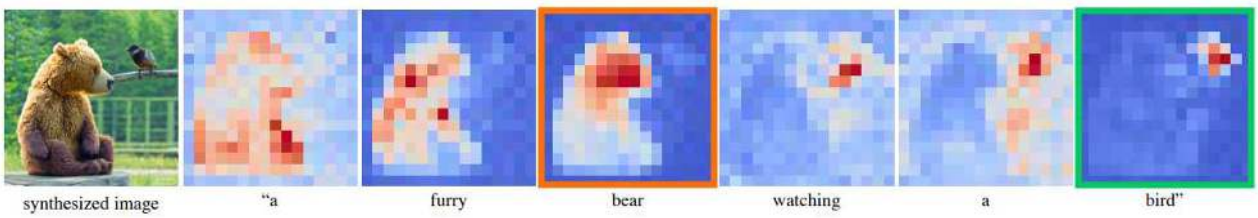

Figure 13 – Attention in Transformer, when using Large Language Models[7].

[7] Hertz, Amir, et al. "Prompt-to-prompt image editing with cross attention control." arXiv preprint arXiv:2208.01626 (2022).

www.pillar-robots.eu

Funded by the European Union