- Subject domain and its corresponding ontology of physical objects — is a child of the Material Entity SD.

The robotic device SD describes the specification of the components of the physical part of the system listed above, and the specification of the actions that the system can perform with or through them.

The SD of physical objects specifies properties of the objects recognized by the system, such as color, shape, volume, weight, etc.

As the system scales up to automate more and more complex processes (line operation, shop floor operation, operation of the entire enterprise), these subject domains will be broken down into subsidiary SDs, e.g. computer vision SDs, robotic arm SDs, etc. Such development of subject domains in the field of production automation has already been done within the framework of OSTIS [9] technology

Figure 10 shows the formalization of the system's knowledge of its own physical part in SCg-code. This formalization is necessary for the system to perform the sorting operation.

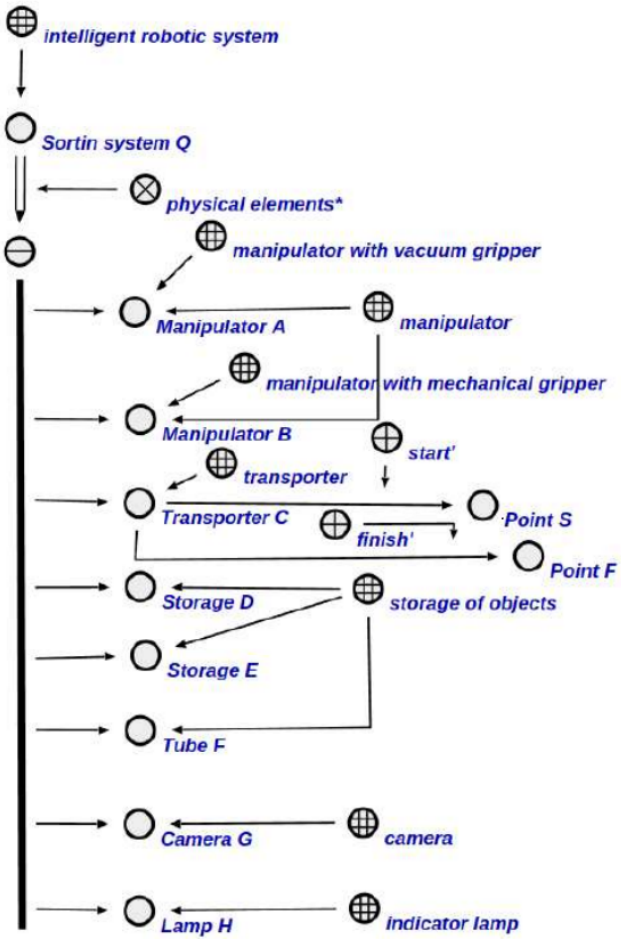


Figure 10 Formalization of the system's knowledge about its own physical part in SCg-code

According to the entity naming convention adopted in the OSTIS [6] technology standard, specific elements of the system are named with a capital letter. For convenience, each specific element is named with a letter:

1) Manipulator A is a sorting arm with vacuum gripper
2) Manipulator B is a feeder arm with mechanical gripper.
3) Transporter C — A transporter having a starting point S as a plurality of objects placed on the transporter by the feeding arm and a point F as a plurality of objects moved by the transporter until the sensor is triggered.
4) The Storage D is a storage of target objects represented by a plurality of target objects in the storage.
5) The Storage E is a storage of objects that are not in Storage D.
6) Tuba F — an object storage represented by a set of unsorted objects.
7) and other.

*C. Program components: problem solver*

The problem solver deals with processing of knowledge base fragments, which is reduced to adding, searching, editing and deleting sc-nodes and sc-connectors of the knowledge base. At the semantic level, such operations are actions performed in the memory of the subject of the action, where, in general, the subject is the system itself, and the knowledge base is its memory.

The specification of an action in the knowledge base describes what should be done, with what, for what, by whom, etc., but the interpretation of actions according to this description is performed by agents. The problem solver of each ostis-system is based on a multi-agent system whose agents interact with each other only through a common knowledge base [10]. Each agent expects some event to occur in the knowledge base. For example, the appearance of a new specification of the action it should perform. When the event occurs, the agent performs the action and places its result in the same knowledge base.

The ostis-system problem solver has methods and tools to divide problems into subtasks and is able to explain its solutions at the level of describing the wording of subtasks and the sequence of their solution. When solving a problem, the problem solver breaks it into subtasks with an explicit description of the wording of each subtask, searches for a method of its solution and applies it.

Partitioning into subtasks begins with analyzing the goal of solving the problem set for the system. Formally, the goal of the developed system can be formulated as follows: for $\forall x$ such that $x \in$ the set of physical objects and $x \in$ the set of objects in Tube F, infer either $x \in$ the set of objects of Accumulator D and $x \in$ the set of target objects, or $x \in$ the set of objects of Accumulator.

E and $x \notin$ the set of target objects. Figure 11 shows the formalization of the goal of the object sorting problem in SCg-code.

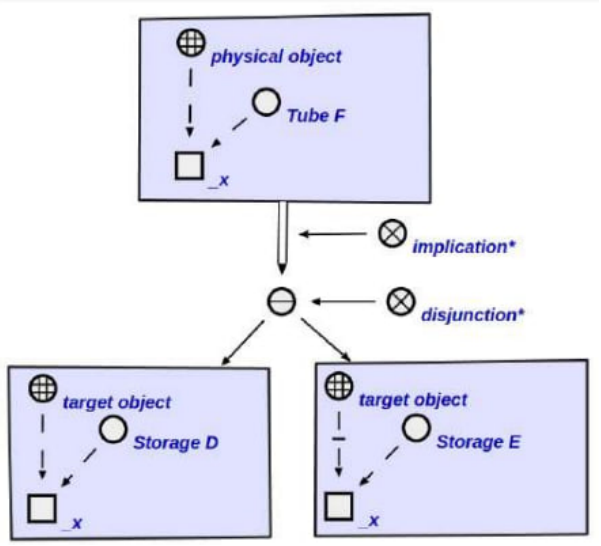


Figure 11. Formalizing the goal of solving the problem of sorting objects in SCg-code

The process of achieving the goal, depending on the system settings, can go in two directions: directly, from the initial situation, and backwards, from the target situation.

Let's consider the case of solution search from the final goal. The task solution search agent breaks the target situation into elementary sc-constructions, so-called triples and fives, and searches the system for actions or logical statements, the application of which can form the necessary part of the target situation in the knowledge base. For example, the specification **of action of moving an object to storage D** states that after its execution the object can get into storage D. However, the initial situation for applying this action is the presence of this object at the point F of the transporter and its belonging to the class of target objects. This initial situation becomes a new target of the solution search agent and the process is repeated for this target. New actions and logical statements are searched for.

Figure 12 shows the specification of the initial and final situation of the action of moving an object to storage D in SCg-code.

In the developed system, the following actions are used to solve the problem of sorting objects (given in the order in which they are found by the problem solving search agent):

1) The action of moving an object to Storage D. An object belonging to the target class and belonging to the set of objects at Point F is moved to the D storage unit. The agent interpreting this action controls Manipulator A.

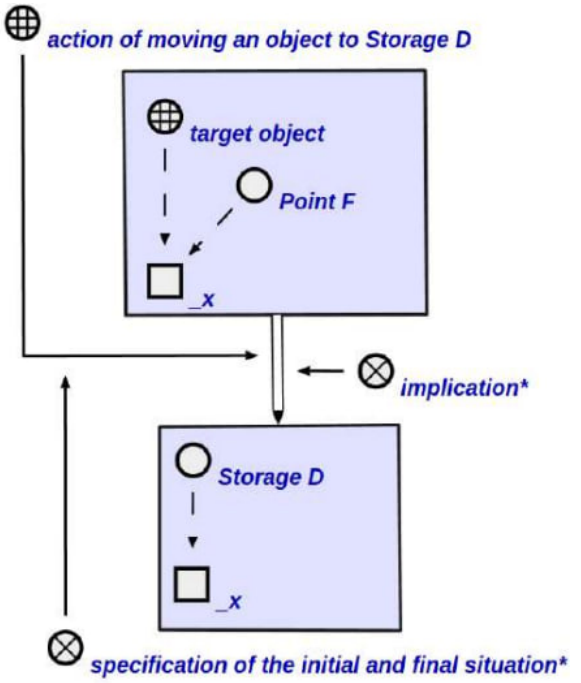


Figure 12. Specification of the start and end situation of the action of moving an object to storage D in SCg-code

2) The action of moving an object to Storage E. An object that does not belong to the target class and belongs to the set of objects at Point F is moved to Storage E. The agent interpreting this action turns on the transporter Storage E for a predetermined amount of time.
3) The action of delivering an object from Point S to Point F. The object belonging to the set of objects at Point S ceases to belong to it and begins to belong to the set of objects at Point F. The agent interpreting this action turns on the transporter until the sensor is triggered.
4) The object classification action. The object belonging to the set of objects at Point F starts to belong or not to the set of objects of the target class. The agent interpreting this action finds a specification of the target class in the knowledge base (see figure 13) and sets this specification as the goal of a new problem solved by the agent according to the principle described above. During the search, the agent applies actions 5), 6) and 7).
5) The action of classifying an object in the image. The class of the object for which its image is specified is determined. The agent interpreting this action uses the methods described in the computer vision module to recognize the object in the image.
6) The action of determining the color of the object in

the image. The agent interpreting this action uses the methods described in the computer vision module.

7) The action of searching for objects in the image. The coordinates in the image obtained from the camera at the moment of sensor triggering are selected for the object. The agent interpreting this action uses the methods described in the computer vision module.
8) The action of moving the object onto the Point S. An object belonging to the set of objects in Tube F is moved to Point S. The agent interpreting this action controls Manipulator B.

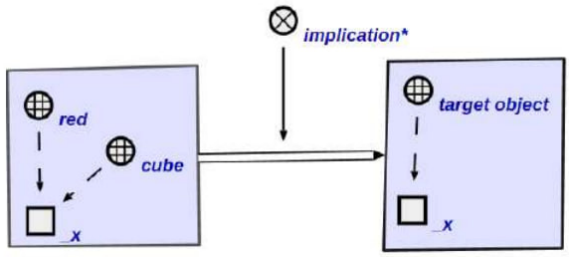


Figure 13. Specification of target class definition in SCg-code

At this point, the agents that interpret these actions use the program interfaces of the physical elements. For example, the agent for moving the object to the accumulator D implements an algorithm that uses the program interface of the manipulator A to transfer to it the rotation angles of its servos for moving the object from the conveyor to the accumulator in rigidly fixed locations. The system development plans include formalization in the knowledge base of the manipulator itself so that the rotation angles of the manipulator servo drives and the order of actions to change them are set depending on the current situation in the working area, rather than according to a predetermined algorithm.

The applied approach allows intelligent robotic systems to solve problems in a declarative way, when the order of application of methods of problem solving is formed by the system independently on the basis of the problem condition. Thus, adding new stages of object processing or methods for determining new characteristics of objects (weight, presence of certain digits, etc.) requires formalization of specifications of initial and final situations of actions and implementation of agents interpreting them, but does not require additional code for integration of these agents. Thus, the design of robotic systems is reduced to the description of the physical part and the tasks to be solved by this system without the need to program the solution of each task.

*D. Program components: computer vision module*

To solve the subproblem of detection of the given objects, the authors have formed a dataset of images of geometric bodies obtained by 3D printing. This sample includes photographs of three geometric bodies (cube, cylinder, cone) taken from different angles and with different camera angles. The total dataset size was 200 images, which were manually labeled. Examples of photos from the dataset, as well as photos from different angles of the same body, are shown in Figures 14 and 15.

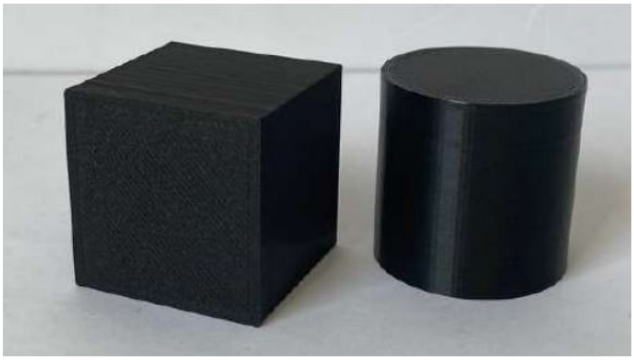

Figure 14. Objects from a dataset of figure images

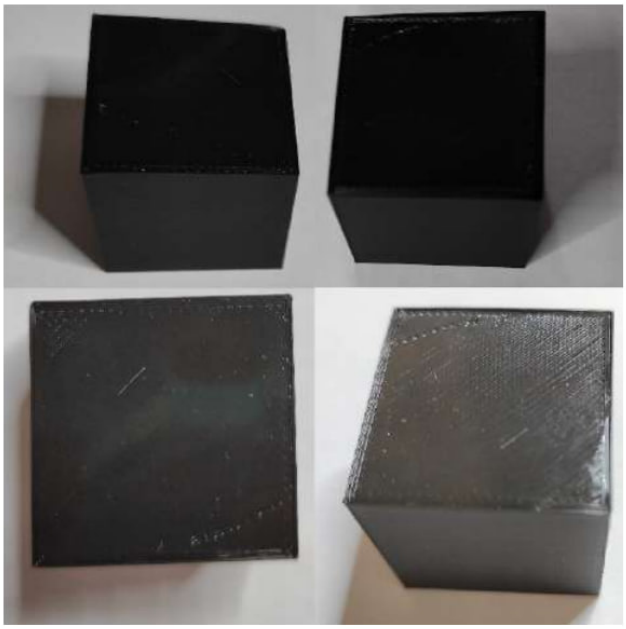

Figure 15. Images of one body taken from different angles

We used a pre-trained YOLO version 7 [11] as a neural network-detector to solve the subproblem of detecting objects of a given type. The whole dataset was split into training and test subdatasets in the ratio of 4:1. After 100 epochs of model training we got $m$AP = **97.4**%.

After training, the model acquired a good level of object recognition, and in some cases was even able to detect objects that were different in color from those in the training dataset (Fig. 16).

Due to the invariance of the model to the color of objects, a simple comparison using the Euclidean metric with a given reference color value was used to determine it.