Chapter 4

Contact event perception

Neuroscience studies presented in Chapter 2, point out that human grasping is driven by the creation and breaking of contacts with the environment. Perception of contact events is performed by humans using multiple sensor cues: visual, tactile, proprioceptive and audio to name a few. Moreover, the prediction of contacts is also an important source of information for error detection and recovery while performing manipulation tasks.

Regarding robots, perception of physical interaction can also be achieved using many sensor modalities: tactile, force-torque, proprioception, accelerometers; even sonar, laser and vision could be used as well. Is a problem in robotics to fuse all the available data to provide the robot with enhanced perception capabilities. Sensor fusion has not been applied so far on contact localization for manipulation.

In this chapter, a sensor fusion framework for contact detection and localization is proposed. It is able to use any knowledge available to detect and localize contacts and improve the robustness and precision of the detected contacts. The presented approach allows the integration of multiple sensors, environment, context and predictions. On top of it, we have implemented a contact event detector that will provide the necessary contact events required by the task description to trigger transitions between states, as shown in Chapter 3.

The framework is divided in three main parts: contact hypotheses generation, hypotheses fusion and contact condensation. Firstly, the contact space and the other basic concepts for the contact hypothesis framework are presented. Secondly, the guidelines for the integration of sensors into the contact space, together with some examples of implementation are shown. Thirdly, the fusion algorithm to combine the different sensory cues and how to perform contact detection and localization from the fusion result is shown. Finally, the approach is validated through several experiments on Tombatossals and a simple use case of a contact driven controller on ARMAR-IIIb. For a detailed description of the robotic platforms used see Appendix A.1 and A.2.

4.1 Related work

In the last decade, contact sensing has become a key element on all the robots with manipulation capabilities. Besides tactile sensors, there are other devices that can measure physical interaction, such as force-torque sensors or joint-torque sensors. Moreover, there are other sensor modalities that can be used like vision or audio [Lacheze et al., 2009]. Unfortunately, each data source has its own representation and the information from different sensors cannot be easily compared with that from other sources. In addition, information about the environment can also be very useful to constrain where physical interaction can occur.

Data fusion from different sources has been a largely studied problem in robotics. In the early 90's the theoretical basis of the current techniques were already settled [Hackett and Shah, 1990]. More recently, the evolution of parallel computation enabled the use of high computational cost probabilistic approaches (e.g. particle filters) [Thrun et al., 2005]. On real scenarios fusion is often performed with a defined goal: fusion of audio and visual input to track a talking person [Fermin et al., 2000], to track an object [Meeussen et al., 2006] or to recognize it [Lacheze et al., 2009]. [Prats et al., 2013] developed a framework that presents sensor fusion for robotic manipulation, where each sensor handles a controller that contributes to the resultant control applied to the robot. In this chapter, instead of focusing on control, we provide a common representation for contact detection and localization.

Using vision and force, [Ishikawa et al., 1996] proposed a method to detect contacts between a known grasped object and the environment. In that work the fusion method is task specific and the contact detection method is embodiment specific. Other works that perform contact localization, either use only one sensor modality [Meier et al., 2011] or process and fuse the data with an ad-hoc non scalable method. [Hebert et al., 2011] presented a probabilistic sensor fusion method to estimate the pose of a grasped object, although they obtained a very good precision (5mm) contacts were considered only on the fingertips.

4.2 Contact hypothesis framework

The sensor fusion framework is composed of two independent parts; the contact hypotheses generators and the integrator (Fig. 4.1). Each generator creates contact hypotheses based on a defined criteria (e.g. force sensor, simulator) and sends them to the integrator. The integrator receives those hypotheses, combines them and uses the result to determine the likelihood of a contact at a location.

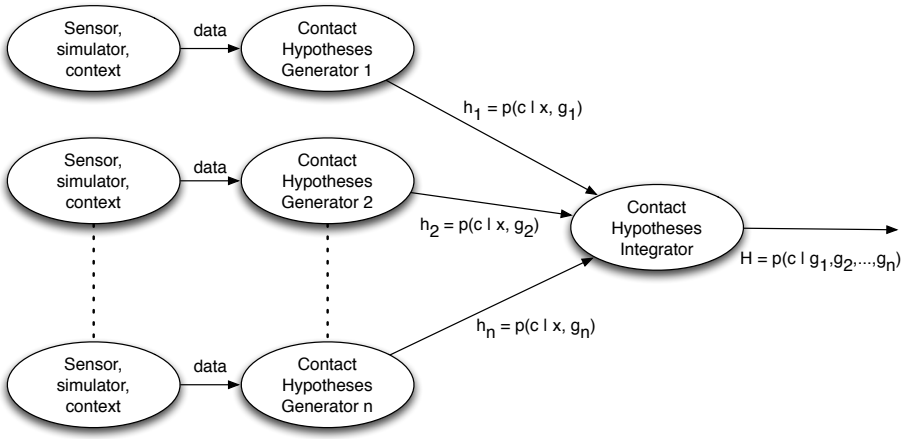


Figure 4.1: System overview. Contact hypotheses generators obtain data from sensors, simulation, kinecmatics, control and environment to produce contact hypotheses that are sent to the integrator. The integrator performs the fusion and contact detection and outputs the resulting contacts.

Contact hypothesis Represents the likelihood that a contact happened at a specified location.

Contact hypotheses integrator Updates the hypotheses space fusing the input clouds of hypotheses from the contact hypotheses generators and provides as a result the estimated contacts.

Contact hypotheses generators Use an information source (context, simulator, sensors) to produce a cloud of contact hypotheses.

Hypothesis space Is a 3D Cartesian space discretized in voxels of a fixed size. The state of the HS is determined by the occupied voxels and the likelihood of each one.

4.2.1 Contact hypothesis and hypothesis space

A contact hypothesis represents the likelihood that a contact happened at a specified location. The hypothesis space HS is a 3D Cartesian space discretized in voxels of a fixed size. The state of the HS is determined by the occupied voxels and the likelihood of each one. The state of the HS is updated by the integrator.

The integrator receives sets of contact hypotheses h_n from the generators $(g_1, ..., g_n)$ that represent the probability p that a contact c happened at a specified voxel $x \in \mathbb{R}^3$. Thus, the set of hypotheses generated by g_n is $h_n(x) = p(c|x, g_n)$ (see Fig. 4.1).

After performing the hypotheses fusion, the output of the integrator is a set of contact hypotheses H representing the probability that a contact happened at a specified voxel combining all the received inputs $H(x) = p(c|x, g_1, ..., g_n)$.

A contact hypothesis must have information about its location and likelihood. Beyond the required information, in the proposed framework, a contact hypothesis is composed by the following elements (required fields are **bold**):

- **Location**: Specifies the 3D position of the hypothesis.
- **Likelihood**: List of likelihoods of each source that contributed to this hypothesis.
- Timestamp: List of timestamps when each likelihood was generated.
- Force magnitude.
- Force direction.
- **Type**: Can take one of this values: Regular, Support or Null.
- **Source id**: List of sources that contributed to this hypothesis.

Contact hypotheses types

It is possible to implement generators that are not based on physical evidence of contacts (e.g. predictions). To avoid detecting contacts without having physical evidence, contact hypotheses are labelled with a type that will be used by the integrator to determine how to proceed for the hypotheses fusion. The possible types are: regular, support and null.

Regular hypotheses

Regular hypotheses are those produced by real sensors from perceptual evidence.

Support hypotheses

Support hypotheses are those produced by generators that do not have perceptual evidence of a contact. Support hypotheses are used to add contextual data or predictions to the estimation of the contact locations. Therefore if the sensors detect a real contact and generate hypotheses, those hypotheses that fuse with support hypotheses will increase their likelihood. On the other hand, support hypotheses that are not fused with any other hypothesis from perceptual evidence will be discarded.

Null hypotheses

Null hypotheses represent the locations of the contact space where there are no contacts. There are cases, where there is relevant information about where contacts cannot happen, this data can be reflected in the contact detection framework generating null hypotheses on the locations where contacts cannot happen. Null hypotheses are used

to draw a null space for the contact detection. Any hypothesis that fuses with a null hypothesis will be discarded.

4.3 Hypotheses generators

The role of a contact hypotheses generator is to convert any suitable information available to the robotic system (context, simulator, sensors) in a cloud of contact hypotheses that will be added by the integrator to the contact space. In Section 4.4 the implementation details of the contact hypotheses generators used in our system are provided. This section gives the guidelines for the implementation of contact hypotheses generators based on sensors and other information sources.

A contact hypotheses generator uses the data from a sensor, a controller, a simulator or any other source that provides information about a perceived or a predicted contact located in the space. The output is a cloud of contact hypotheses (that determine the possible contact locations), the likelihood of each generated contact hypothesis and its type.

4.3.1 Implementation guidelines

For the implementation of a new generator, the first step is to determine whether the generator can detect one or more contacts at a time. On the one hand, single-contact generators can only detect one contact at a time (e.g. bumpers or force sensors). On the other hand, multi-contact generators are able to detect multiple contacts simultaneously (e.g. tactile sensor arrays, simulation engines).

For single-contact hypothesis generators, the probability of the detected contact has to be distributed among the generated contact hypotheses, see Eq.(4.1). The likelihood can be distributed uniformly or with any other distribution, depending on the nature of the data used. For multi-contact hypothesis generators, we will allow the sum of likelihoods to be equal or greater than one, see Eq.(4.2), because more than one contact can be detected at a time.

$$\sum^{i} p(c|x_i, g_{single}) = 1 \tag{4.1}$$

$$\sum^{i} p(c|x_i, g_{multi}) \geq 1 \tag{4.2}$$

The second step is to calculate the likelihood of each generated contact hypotheses. To do so, the nature of the data that the generator uses has to be identified. Although there may be more types, in this thesis the following types were identified:

- Binary: (contact / no contact) it gives no clue about the contact distribution, in that case a fixed value for all the hypotheses is used. The constant value can be determined by the sensor model or experimentally probing the sensor and determining the likelihood of a contact to be detected. A bumper or a deterministic simulator are examples of binary data sources that can provide information about the existence of a contact but not a related value. See implementation examples in Sec. 4.4.3 and Sec. 4.4.5.

$$p(c|x, g_{binary}) = constant \tag{4.3}$$

- Value: If the value of the input data is directly related to the contact likelihood. The final probability for each hypothesis can be calculated using Eq.(4.4) where $argmax(data)$ is used for normalization and represents the maximum value of the current reading only if $argmax(data) > 0$. This approach assumes that the sensor does not provide false positive readings, otherwise if the sensor output is different from zero when there is no contact (e.g. due to noise or hysteresis effects) those values will be considered with high likelihood, in this cases the sensor input has to be previously filtered. An example of this kind of data are tactile or pressure sensors, in those cases the higher the value the more likely that there is a contact on the location where the sensor is at. An implementation example is provided in Sec. 4.4.2.

$$p(c|x, g_{value}) = \frac{data_x}{argmax(data)} \tag{4.4}$$

- Distance: If the data used to generate hypotheses is a distance (e.g. from the sensor to an object). An inverse square law like Eq.(4.5) can be used to determine the likelihood of each hypothesis. Where λ determines the distance at which the likelihood will be 0.5, this has to be tuned depending on the precision of the sensor and its calibration. An implementation example is provided in Sec. 4.4.4.

$$p(c|x, g_{distance}) = \frac{\lambda^2}{\lambda^2 + distance^2} \tag{4.5}$$

Finally, the last step is to determine where in the contact space the hypotheses will be generated. If the location of the hypotheses can not be determined by the data used to generate the hypotheses, other information available can be used for that purpose such as the sensor's geometry and location, robot geometry, joint configuration, sensor sensitive area, etc.

For example a bumper based contact hypothesis generator, would be single-contact and will use the sensor geometry to place the contact hypotheses on. On the other hand, a LiDAR combined with the robot geometry would be multi-contact, use the robot

geometry close to the range data to place the hypotheses and use the distance between the range data and the robot model as the likelihood of each contact hypothesis. Some more examples are given in Section 4.4 where the implemented hypotheses generators for our perceptual system are detailed.

Summarizing, to implement a new contact hypotheses generator: first, the location where the hypotheses will be generated has to be determined. Second, the type of the generator (single-contact or multi-contact) has to be specified. Finally, the type of data and the law for the generation of each hypothesis likelihood has to be selected.

4.4 Implemented regular hypotheses generators

In this section, we show the implementation of several contact hypotheses generators based on sensors that are often available in robotic manipulators.

The sensors embedded on nowadays robots, are usually platform dependent and require a specific processing depending on each embodiment. In this chapter we have only proposed the implementation of platform dependent hypotheses generators. However, the guidelines for the implementation of other types of contact hypotheses generators are given in Section 4.3. On the other hand, the implementation of the integrator is platform agnostic.

4.4.1 Experimental platforms

We have implemented contact hypotheses generators for two robotic platforms, Tombatossals and ARMAR-IIIb.

Tombatossals, is a humanoid torso with 29 DOF with advanced sensing capabilites such as tactile sensors, force-torque sensors, cameras, and a kinect, detailed information about this robot is provided in Appendix A.1. ARMAR-IIIb is a humanoid robot with 33 actuated DOFs. For the experiment we have only used its right arm (7DOF) and hand (7DOF). It has a force-torque sensor on the wrist and tactile sensor pads on the palm and fingertips. More details about this robot can be found in Apendix A.2.

4.4.2 Tactile sensor hypotheses generator

One of the main sensory cues that can provide information about contacts between the robot and the environment are tactile sensors. This kind of sensors typically produce an array of pressure values with measurements from a grid of sensing cells. In combination with the joint positions and the robot model it is possible to determine the spatial location of contacts.

If there are multiple contacts at a time, tactile sensors are able to detect them and provide sensor readings accordingly, hence this will be a multi-contact generator. In this kind of sensors, a single contact may generate marginal readings in the nearby taxels

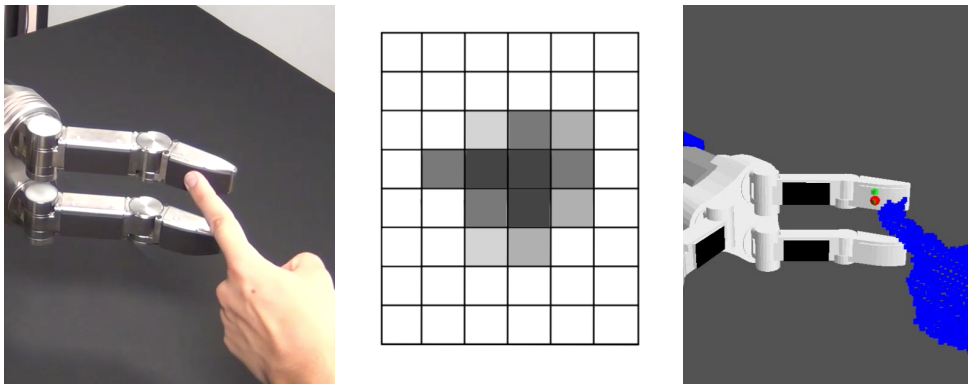


Figure 4.2: Example of the hypotheses generated by the tactile sensor generator. Left: Real scenario. Center: Tactile sensor readings. Right: Green voxels show generated hypotheses and red sphere the result of contact localization.

(see Fig. 4.2 center). However, the contact point will have a higher pressure value, thus we consider that the taxel pressure value is related with the contact likelihood at that taxel.

The hypotheses location x is determined by the activated taxels of the sensor, see Fig.4.2. The likelihood of a hypotheses located at a voxel x and with a tactile value of z_x is calculated using the following equation:

$$p(c|x,z) = \frac{z_x}{argmax(z)} \tag{4.6}$$

The tactile hypotheses generator was implemented both for ARMAR-IIIb and Tombatossals.

4.4.3 Force-torque hypotheses generator

Nowadays, service robotics oriented manipulators, often provide the external forces and torques applied to their end effectors. A straightforward solution to provide information about contacts is to use the force-torque readings. That data can be obtained from a force-torque sensor or computed using the joint efforts and the robot dynamic model. This sensory data is very valuable for contact detection because it can provide information about physical interaction.

In this subsection we present an implementation of a contact hypothesis generator based on the force-torque data detected at the end effector.

As explained in Sec. 4.3, the first step to implement a new hypotheses generator is to determine its type. It is not possible to determine the location of multiple contacts with

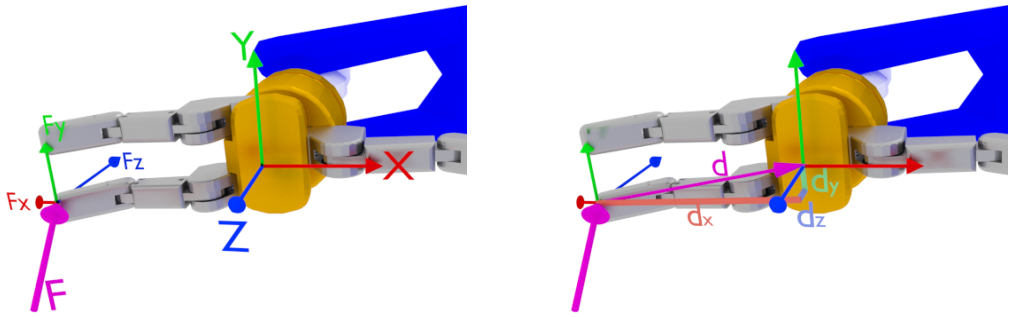


Figure 4.3: Left: A force F is applied to the finger of the robot, that force could be applied by an external actor or be a reaction to the robot interaction with the environment. The sensed force is decomposed in f_x, f_y and f_z by the force sensor. Right: The perceived torque τ depends on the distance vector d where the force F is applied from. Each component of the vector τ can be obtained from the sum of the two tangential forces.

a F/T sensor. Hence, this is a single-contact hypothesis generator because it can only detect one contact at a time.

Regarding the data obtained from the sensor, the values provided give no information about the contact likelihood, thus the likelihood of the detected contact will be uniformly distributed among all the generated hypotheses to satisfy Eq. (4.1).

In order to determine the location where the contact hypotheses will be generated, we have to take a look on how the force-torque sensor works. These type of sensors produce a 3D force vector $f \in \mathbb{R}^3$ and a 3D torque vector $\tau \in \mathbb{R}^3$. The perceived torque τ depends on the distance vector $d \in \mathbb{R}^3$ where the force F is applied from (see Fig. 4.3 right). Each component of the vector τ can be obtained from the sum of the two tangential forces that act on that component, see Eq.(4.7).

$$\tau_3 = f_{\perp 1} \cdot d_{\perp 2} + f_{\perp 2} \cdot d_{\perp 1} \tag{4.7}$$

Applying Eq.(4.7) for each of the axes, we obtain the system of equations shown in Eq.(4.8).

$$\begin{cases} \tau_x = f_z \cdot d_y - f_y \cdot d_z \\ \tau_y = f_x \cdot d_z - f_z \cdot d_x \\ \tau_z = f_y \cdot d_x - f_x \cdot d_y \end{cases} \quad \text{w.r.t. sensor frame} \tag{4.8}$$

Solving the linear equation system with 3 equations and 3 unknowns shown in Eq.(4.8), will provide the contact point ($d = [d_x d_y d_z]$). Unfortunately, the equations are linearly dependent and the system has no single solution. However, the result of intersecting the three equations is a line that can be used to generate contact hypotheses.

In order to generate a set of possible solutions, we will use only two of those equations at a time (see Eq.(4.9)) and solve the leftmost group for a set of possible values for d_x (if $f_x \neq 0$), the center group for d_y (if $f_y \neq 0$) and the rightmost group for d_z (if $f_z \neq 0$). To select the set of values used for d_x, d_y and d_z, the bounding box of the hand is expanded and used to limit the sampling space. After this, we have three lines of contact hypotheses, see Fig.4.4 center. The resulting lines are theoretically equal but experimental validation has shown that the resulting lines are not always equal, it might be due to sensor noise and may depend on the nature of the sensor.

$$\begin{cases} d_y = \frac{f_y \cdot d_x - \tau_z}{f_x} \\ d_z = \frac{\tau_y + f_z \cdot d_x}{f_x} \end{cases} \quad \begin{cases} d_x = \frac{\tau_z + f_x \cdot d_y}{f_y} \\ d_z = \frac{f_z \cdot d_y - \tau_x}{f_y} \end{cases} \quad \begin{cases} d_x = \frac{f_x \cdot d_z - \tau_y}{f_z} \\ d_y = \frac{\tau_x + f_y \cdot d_z}{f_z} \end{cases} \tag{4.9}$$

Finally, using the distance of the generated hypotheses to the spherical model of the robot (details about the robot spherical model are presented in Appendix B), those hypotheses that are not close to the robot geometry will be filtered out (See Fig.4.4 right). The likelihood of the detected contact will be uniformly distributed among all the generated hypotheses, therefore the likelihood of a hypothesis will be $p(c|x, g_f) = 1/n$ where n is the number of generated hypotheses.

Another approach to determine the contact point regardless of end effector geometry is shown in [Karayiannidis et al., 2014]. The Force-torque hypotheses generator was implemented both for ARMAR-IIIb and Tombatossals.

4.4.4 Range sensor hypotheses generator

Tactile sensing technologies, provide accurate data about contacts and are very sensitive. However, it is not possible to cover the whole robot hand with tactile sensors and usually there is a lot of surface that cannot sense contacts using this modality. That gap is covered by force-torque sensors which can sense contacts occurring at any point of the end effector. Unfortunately, their sensitivity is not good enough to detect contacts with light objects like an empty cardboard box. As a consequence, even in heavily sensorized systems, it is common that a contact goes unnoticed which jeopardizes the performance of a contact-based manipulation system.

This subsection proposes a contact hypotheses generator based on range data provided by an RGBD active camera. This generator can be used to fill in the gap left by tactile and force-torque sensors regarding light objects and improve contact detection

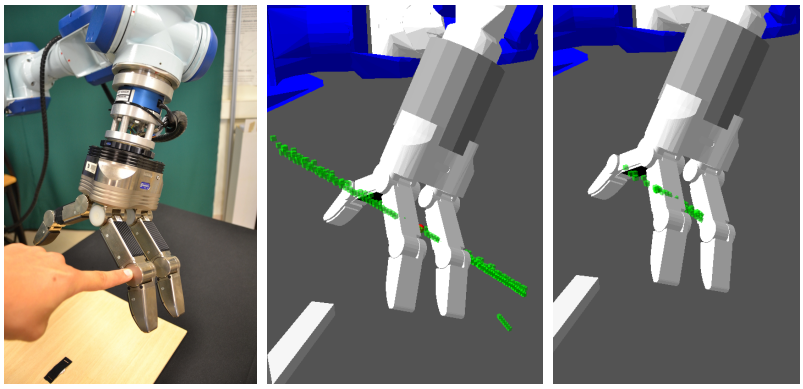


Figure 4.4: Example of the hypotheses generated by the force-torque generator. Left: Real contact. Center: Generated hypotheses before hand geometry filtering. Right: Generated hypotheses.

and localization on those situations. However, the implementation of a method able to detect contacts using an RGBD camera, is not as straightforward as using a sensor that directly measures physical interaction.

One of the problems to address is that contacts can occur on non visible parts of the object, or most likely be occluded by the robot itself. To deal with the occlusion problem, this contact hypotheses generator utilizes an Occupancy Grid Map (OGM) in order to determine where collisions can happen (Fig. 4.5).

The steps performed by the range sensor hypotheses generator are depicted in Fig. 4.6. 1) Before the manipulation task starts, the OGM of the scene is initialized. 2) While the robot moves to manipulate the object, the OGM is updated using the robot spherical model and the space traversed by the robot is removed from the possible contact locations. At the same time, the object is tracked to detect its movement. 3) Once object motion is detected, the contact location is estimated using the combination of the OGM and the spherical model of the robot (details about the robot spherical model are presented in Appendix B).

The generator assumes that if the manipulated object moves, the motion is caused by the physical interaction of the robot, no external agents are acting on the environment.

OGM initialization

The OGM is initialized by projecting each point of the initial object point cloud along the direction of the camera until it intersects with the table plane. To project the points, a perfect pin-hole model of the camera is considered. The initial object point cloud is obtained before any manipulation action is taken. To detect the table and segment out

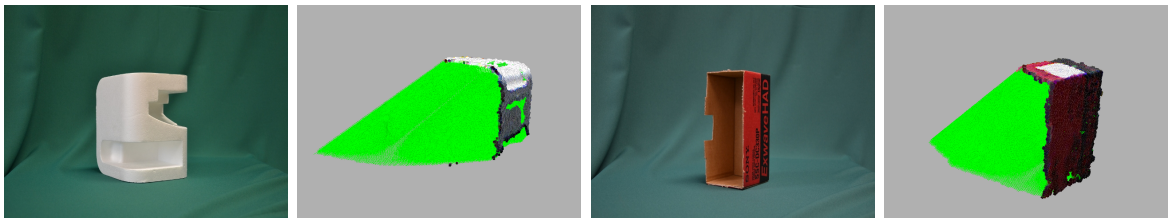


Figure 4.5: Real objects and its initial Occupancy Grid Map (OGM).

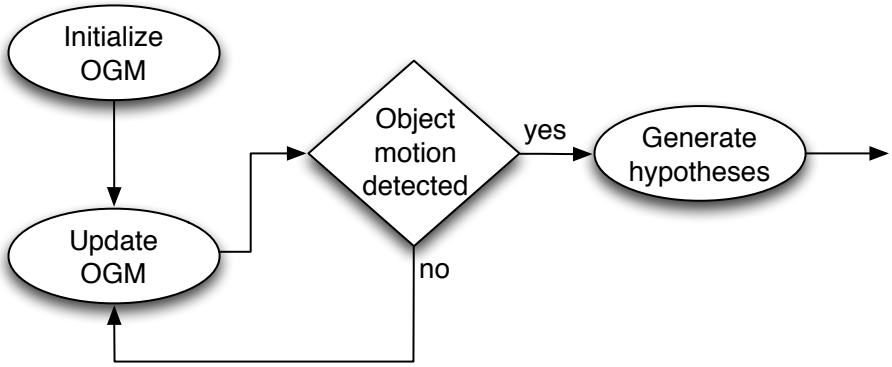


Figure 4.6: Range hypotheses generator diagram. After initializing the OGM, the algorithm keeps updating the OGM until object motion is detected. Then the contact hypotheses are generated.

the object, the same visual pipeline used in the experiments presented in Chapter 3 is used. The details of the visual pipeline are given in Section 6.3.3.

Then, the occluded area is discretized in voxels of $1mm^3$ in each direction (x, y, z). The voxels that are already present in the point cloud provided by the RGBD sensor, have a probability of being occupied $P(c_i) = 1$. Meanwhile, there is no information about the voxels that are not being seen, hence its starting likelihood of being occupied is $P(c_i) = 0.5$. The initial state of the OGM is depicted in Fig. 4.5 where the green voxels correspond to occluded space and the color voxels correspond to voxels that are seen by the camera and are occupied.

Virtual contact sensor model

To update the OGM a probabilistic sensor model is required. As there is no real sensor that can provide readings about contact likelihood, a virtual sensor based on the robot spherical model is used. The model provides the probability function of having a contact z given a voxel state c_i. The function is defined depending on the distance between the robot model and the voxel d, see Eq. 4.10.

$$P(z|c_i) = \begin{cases} 0.0 & \text{if } d < 0 \\ 0.5 & \text{if } d > 0 \\ 0.9 & \text{otherwise} \end{cases} \tag{4.10}$$

If the voxel is inside the model then $P(z|c_i) = 0$, we can be sure that the voxel is free of contact. If the voxel is outside the hand model, there is no information about the contact state of that voxel, hence $P(z|c_i) = 0.5$. Finally, if the voxel is on the surface of the hand model it is likely that the contact happened there, thus $P(z|c_i) = 0.5$.

Object motion detection

To detect the motion of the object produced by a contact, an object tracking approach is implemented. In the literature there are algorithms for model-less object tracking based on image descriptors such as SIFT [Lowe, 1999], SURF [Bay et al., 2008], or Harris corners [Harris and Stephens, 1988]. However, these methods make the assumption that the objects have texture, a regular shape or that the descriptors are visible all the time. As we do not make any assumption of shape or texture and the hand of the robot may occlude parts of the object, we have chosen the Iterative Closest Point (ICP) algorithm [Zhang, 1994] as our approach to object tracking. In particular, we use the standard ICP algorithm implemented in PCL [Rusu and Cousins, 2011]. The ICP does not require any features of the object and exploits 3D data.

ICP is applied to the object point cloud at instant t^k and t^{k-1}. The algorithm provides as a result a homogeneous transformation matrix that is decomposed in a translation

vector $T(x, y, z)$ and a quaternion $q(x, y, z, w)$ from which an angular rotation Θ is obtained.

When the magnitude of the translation vector $\|T\|$ or the angular rotation Θ are grater than a configurable threshold ($\|T\| \geq T_{max}$ or $\Theta \geq \Theta_{max}$), the object has moved and therefore a contact has happened. The thresholds should be tuned regarding the noise of the sensor used and the noise introduced if the hand occludes the object.

OGM update

The OGM is updated each time step. If no object movement is detected, the voxels that are inside or on the surface of the model are updated. The new likelihood value for this voxels is 0, as can be obtained replacing $P(z|c_i) = 0$ in Eq. 4.11. Voxels that are outside the model and not on the surface, keep the same value. Finally, if object movement has been detected the surface voxels are updated according to the following rule:

$$P(c_i|z)^k = \frac{P(c_i|z)^{k-1} \cdot P(z|c_i)}{\sum_{c_i} (P(c_i|z)^{k-1} \cdot P(z|c_i))} \tag{4.11}$$

Where $P(z|c_i)$ is the probability of a measure given a occupied voxel (c_i), in other words, the virtual sensor model. $P(c_i|z)^{k-1}$ is a priori probability to be occupied given a measure z and $P(c_i|z)^k$ is the a posteriori probability to be occupied given a measure z.

Contact hypotheses generation

When object movement is detected, contact hypotheses are generated. First, the voxels from the OGM with high likelihood are selected. Second, contact hypotheses are generated on the location of those selected voxels. Finally, the likelihood assigned to the hypotheses is uniformly distributed among all the generated hypotheses.

The range sensor hypotheses generator was implemented only for Tombatossals, although it can be implemented also for ARMAR-IIIb using its stereo head as a range data sensor.

4.4.5 Finger pose feedback hypotheses generator

This generator exploits the compliance of robotic hands. Usually, contacts on compliant fingers will move the finger joints without any control command being applied to them. The involuntary variation of compliant hand finger positions can be detected by the hand encoders and used to detect a contact on the fingers.

If there are multiple contacts at a time on the same finger, the encoders are not able to detect that situation. Thus, this is a single-contact hypotheses generator. To determine

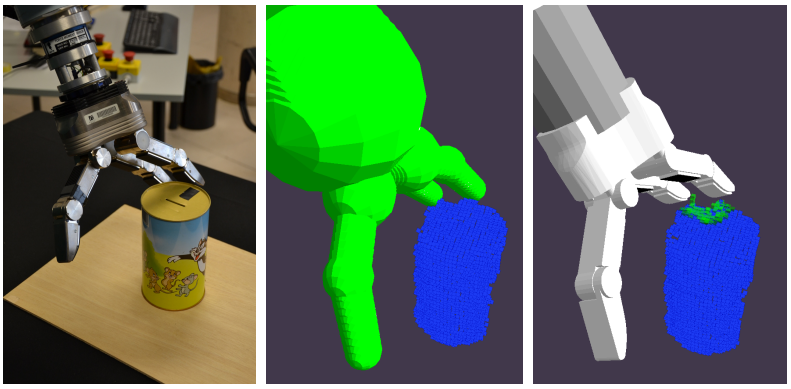


Figure 4.7: Example of the hypotheses generated by the range sensor generator. Left: Real scenario. Center: Segmentation using spherical model. Right: Generated hypotheses.

the location where the contacts will be generated, as there is no information available on the direction or position of the contact, it has to be inferred from the joint readings. Using the robot model, the contacts will be generated all over the finger's surface.

Before starting the detection, the current pose of each finger joint is stored. When a variation on a joint is detected, the finger geometry together with the proprioception and joint values is used to place the contact hypotheses. Like for the force-torque generator, the likelihood is uniformly distributed among all the generated hypotheses, see Fig. 4.8.

This generator assumes that the hand joints are not actuated, if the hand is moving or actively applying forces to the environment this generator is disabled. After moving the hand joints to a different position, the generator is reset to get the reference values updated.

The finger pose feedback hypotheses generator was implemented only for ARMAR-IIIb. Tombatossals does not have compliant hands.

4.5 Implemented support hypotheses generators

In this section we present the implementation of support contact hypotheses generators based on simulation, kinematics and control. It is important to remind that, as explained in Sec. 4.2.1, *Support* hypotheses cannot be considered standalone, *Support* hypotheses that are not fused with *Regular* hypotheses will be discarded. The final effect of *Support* hypotheses is to narrow down the contact localization projecting predictions into the *Regular* hypotheses provided by other generators. The use of *Support* hypotheses, allows us to project predictions or beliefs into the contact space. The benefits of using this approach are experimentally validated in Section 4.7.

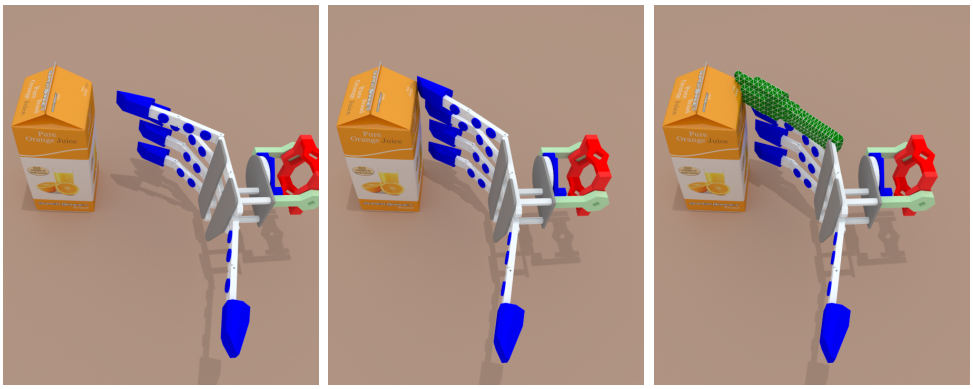


Figure 4.8: Example of the hypotheses generated by the finger pose feedback hypotheses generator. Left: Initial scene, hand moving towards the object. Center: The hand contacts the object and the compliant finger bends back. Right: Generated hypotheses on the compliantly bended finger.

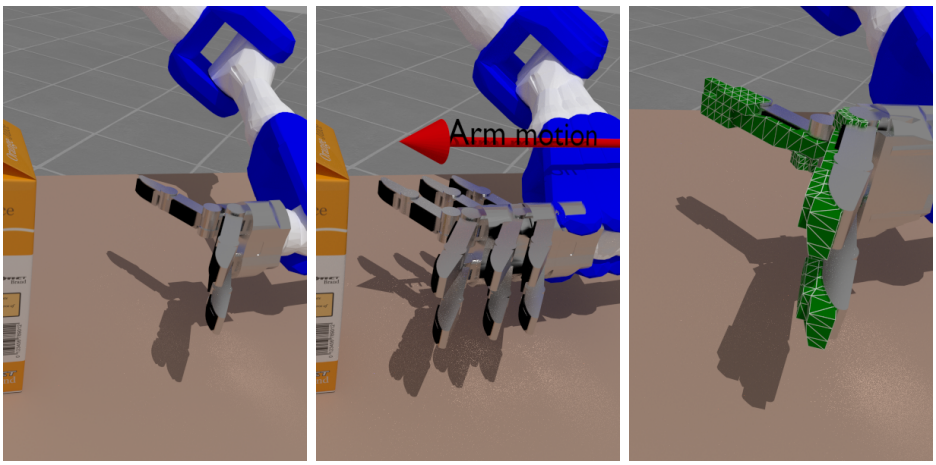


Figure 4.9: Example of the hypotheses generated by the motion estimation support hypotheses generator. Left: Initial scenario. Center: Arm motion Right: Generated support hypotheses.

4.5.1 Motion estimation support hypotheses generator

This generator exploits the current motion of the robot to produce support contact hypotheses in the locations where the robot will be in the next time step. Using the current joint positions and their velocity, contact hypotheses are generated on the next predicted hand position. As this generator is not based on physical evidence, it produces only *Support* hypotheses.

This generator estimates the position of the robot joints in the next time step using the previous joint positions. The predicted position of the robot joints $q(t+1)$ is calculated using Eq.(4.13)

$$\Delta q = q(t) - q(t-1) \tag{4.12}$$

$$q(t+1) = q(t) + \Delta q \tag{4.13}$$

With the predicted joint positions, the generator uses the robot model to produce support contact hypotheses on the space that will be occupied by the robot in the next time step as depicted in Figure 4.9.

There are no constraints about the number of contacts that could be detected using this method, thus this is a multi-contact generator. Moreover, there is no value that can be related to the contact likelihood and the hypotheses are generated with a fixed likelihood value. The likelihood of the generated hypotheses depends on the weight we want to give to this support generator. During the experiments we found that a good value is 0.3. Motion estimation support hypotheses generator was implemented both for ARMAR-IIIb and Tombatossals.

4.5.2 Simulator predictions support hypotheses generator

Another opportunity to narrow down the contact localization problem, is the use of simulators to estimate where the contacts are more likely to happen. Using the simulator as a prediction engine, together with the model of the environment and the objects, allows us to generate support hypotheses at the locations where the simulation engine detects collisions between the robot and the environment or objects.

This generator produces *Support* hypotheses because it is not directly measuring physical interaction between the robot and the environment. It uses the integrated OpenGRASP simulator as a prediction engine to detect where contacts are supposed to happen. In the simulator, the model of the robot, the environment and the objects in the workspace are previously loaded. The simulator implementation as a prediction engine is detailed and discussed in detail in Chapter 5.

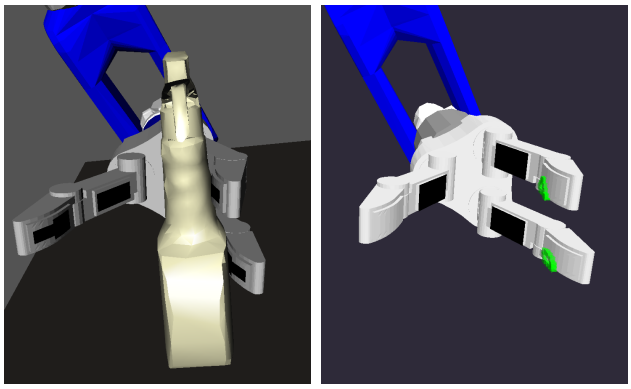


Figure 4.10: Simulator prediction support contact hypotheses generator. Left: Simulator, Right: Support contact hypotheses generated (green voxels).

This generator uses the available methods in OpenRAVE/OpenGRASP and generates *Support* hypotheses where contacts in simulation are detected (Fig. 4.10). There is no limitation on the number of contact points that this generator can detect, thus this is classified as a multi-contact generator. Moreover, the set of contacts provided by the simulation engine do not have any related confidence value that can be used as the likelihood. Hence, the likelihood used for each generated support contact hypotheses, depends on the weight that the simulator will have in the contact localization process. While performing our experimental validation, we found a good value on 0.5. This generator is only implemented for Tombatossals, it can be implemented on ARMAR-IIIb as well using its integrated simulation environment Simox [Vahrenkamp et al., 2012].

4.6 Contact hypotheses integrator

The contact detection and localization process is separated in two main steps. First, the hypothesis space HS is updated fusing the multiple inputs from the different contact hypotheses generators. Finally, the HS is traversed and a contact condensation method is applied to determine the estimated contacts that will be the output of the whole contact detection framework.

4.6.1 Hypotheses fusion

After all the hypotheses generators have provided their clouds of contact hypotheses, the fusion process is performed by the integrator. The integrator receives contact hypotheses from any number of generators, then fuses the incoming hypotheses and produces estimations of contact locations (Fig. 4.11).

As introduced in Sec. 4.2.1, the fusion is performed in the hypothesis space HS. At the beginning, the hypothesis space is empty. When a cloud of contact hypotheses is received by the integrator, the hypotheses are processed one by one and added to the hypothesis space.

When adding a new hypothesis, the integrator uses the hypothesis location to check whether that voxel is already occupied by a hypothesis. If so, both hypotheses are fused, otherwise the hypothesis is inserted into the voxel. When two hypotheses are fused, the resulting hypothesis keeps the location of the voxel and the force and direction are averaged using both hypotheses values weighted by their likelihood. The lists of likelihoods, timestamps and sources are combined with the incoming lists keeping the newest value if the same source is in both the incoming and current lists. A hypothesis is discarded when all its timestamps are older than a configurable timeout parameter. This parameter should be adjusted depending on the update rate of the hypotheses. The fusion process is detailed in Alg 4.1.

It is possible that the incoming hypotheses are generated from different sources at different rate, to perform the hypothesis fusion, the integrator waits until the hypotheses of all the active sources are received. From faster sources, the newest readings are used. The sources can be plugged and unplugged to the integrator dynamically.

After receiving and combining the contact hypotheses from all the active sources, the fused likelihood of each occupied voxel is computed using the DeMorgan's law, see Eq.(4.14). We assume that the measurements of the sensors are independent of each other.

By combining the likelihoods using Eq.(4.14) we expose the integrator to be saturated by inputs like $p(c|x, g_n) = 1$. On the other hand, the saturation will occur only on determined voxels and will not affect the entire HS. Moreover, the design of contact hypotheses generators should take into account this issue, and produce very high likelihoods only when necessary.

$$p(c|x, g_1, ..., g_n) = 1 - \prod_{i=1}^{n}(1 - p(c|g_i)) \tag{4.14}$$

4.6.2 Contact condensation

As a result of the hypotheses fusion, the hypotheses space contains a cloud of contact hypotheses with different likelihoods (see Fig. 4.11 center right). This contact hypotheses provide information about the possible contact locations, in order to provide the estimated location of the contacts, the cloud of contact likelihoods is post-processed. To obtain the estimated contacts and their location from the fused hypotheses cloud, different methods can be used:

Algorithm 4.1 Contact hypotheses fusion algorithm

```
function ProcessIncomingHypotheses(hypotheses)
    for all h in hypotheses do
        if IsVoxelOccuppied(h.location) then
            FuseHypotheses(h, GetVoxel(h.location))
        else
            SetVoxel(h)
        end if
    end for
end function
function FuseHypotheses(h1, h2)
    h3.location := h1.location
    h3.likelihood := h1.likelihood ∪ h2.likelihood
    h3.sources := h1.sources ∪ h2.sources
    h3.timestamp := h1.timestamp ∪ h2.timestamp
    h3.fMagnitude := weightedMean(h1.fMag, h2.fMag)
    h3.fDirection := weightedMean(h1.fDir, h2.fDir)
    SetVoxel(h3)
end function
```

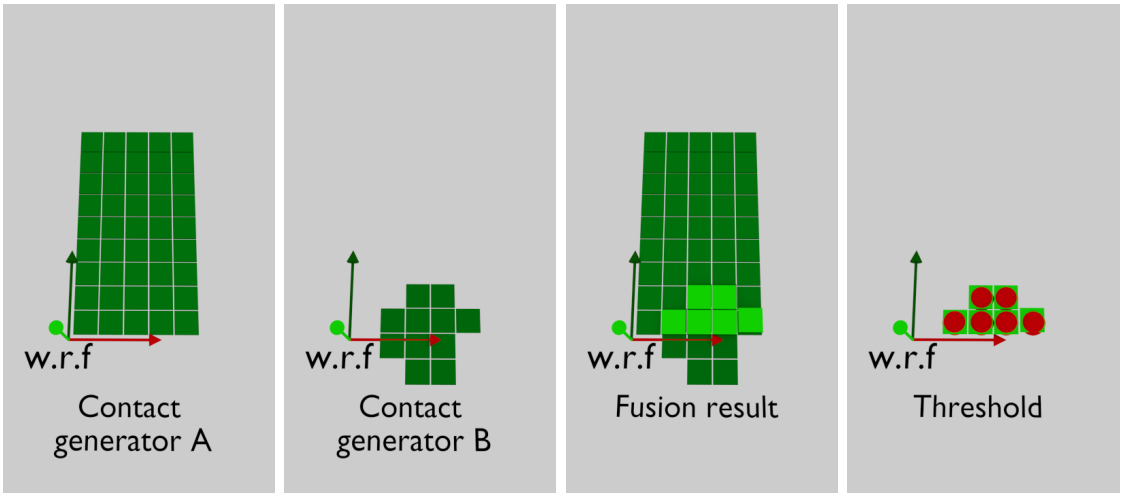


Figure 4.11: Contact hypotheses fusion and contact detection using the threshold method for contact condensation. The likelihood of each hypothesis is color encoded (Dark green: low probability, Light green: High probability). Red spheres show the result of the contact condensation step. Note that all the pictures use the same reference frame.

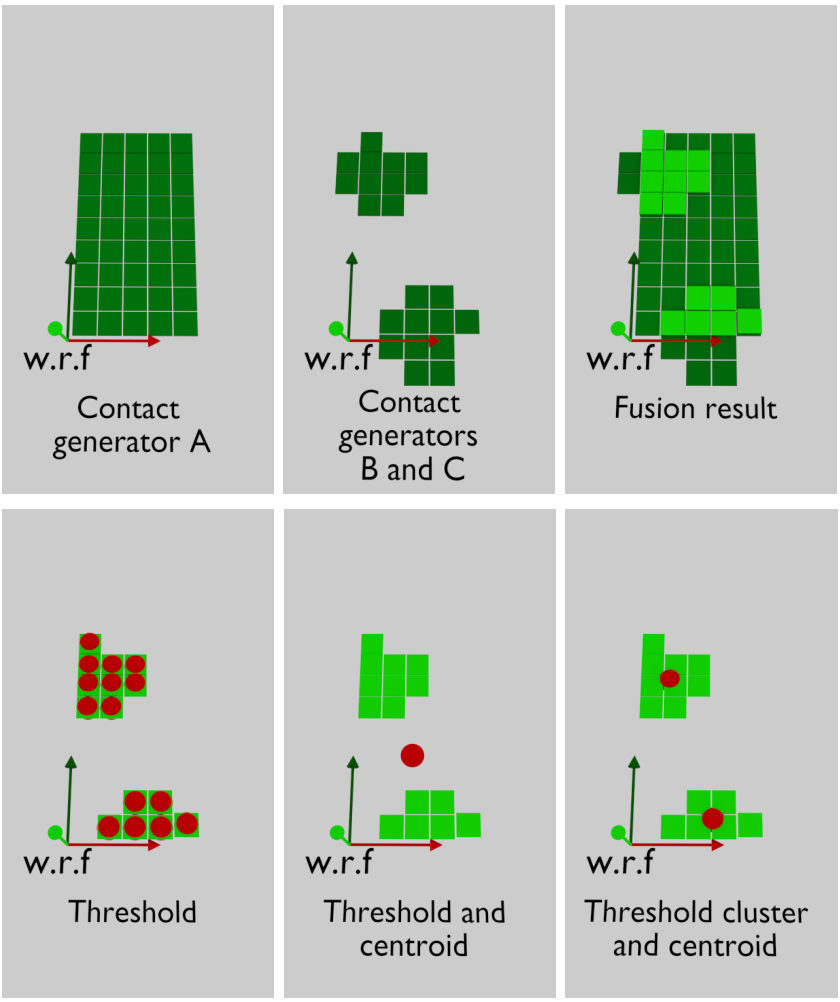


Figure 4.12: Comparison of the contact detection results using three different contact condensation methods: threshold, weighted centroid and threshold-cluster-centroid. Red spheres show the result of the contact condensation step. In this case the clustering based method is able to detect the two contacts accurately while the weighted centroid detects only one and the threshold detects too many.

- Threshold: threshold the hypotheses using their likelihood.
- Centroid: calculate the centroid weighted by the likelihood.
- Clustering: perform clustering to detect the connected hypotheses.

Those operations can be applied to the result of the fusion process. Depending on the order of application and the parameters used, different results will be obtained. We have explored the application of a simple threshold and a more sophisticated approach that performs a sequence of threshold, cluster and centroid operations. The following subsections give more details about the parameter settings and the results obtained using this approaches for contact condensation.

Threshold

A straightforward method is to set up a high threshold for the likelihood (e.g $H(x) \geq 0.6$), and filter the data to obtain the hypotheses that will be considered contacts. It is likely that the result is a cloud of contacts around the real location (see Fig. 4.11), to provide a single contact after applying the threshold, a centroid calculation weighted by the hypotheses likelihood can be calculated.

This strategy is eligible if all the generators produce very precise data, otherwise the likelihood will be distributed among many hypotheses and none of them will be over the threshold.

Threshold, cluster and centroid

On the other hand we can use a low threshold (e.g $H(x) \geq 0.1$) and calculate the global centroid weighted by likelihood. As in the previous method this will reduce the detected contacts to a single one, if there are separated contact regions the result will be the average of those regions. Thus, before performing the centroid calculation, the contact regions are detected using an euclidean clustering algorithm[1]. Then the centroid (likelihood weighted) is calculated for each cluster (See red spheres in Fig. 4.12).

With this condensation method, the system is able to detect multiple contacts at a time and localize them more precisely instead of providing a cloud of contacts. On the other hand the clustering method computation requires a nearest neighbour check for almost every hypothesis, and in high cluttered hypothesis spaces it can reduce the speed performance. A comparison of different combinations for the contact condensation is depicted in Fig. 4.11.

[1]Clustering algorithm taken from: `http://www.pointclouds.org/documentation/tutorials/cluster_extraction.php`

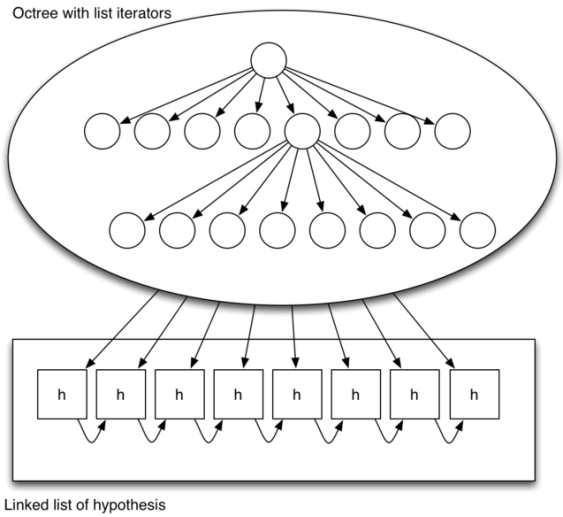


Figure 4.13: Suggested data structures for the implementation of the contact space and the sensor fusion framework. An octree indexes the linked list that contains all the contact hypotheses.

4.6.3 Implementation guidelines

The contact hypotheses space occupation is represented by an octree to efficiently retrieve whether a voxel is occupied by a hypothesis. The core implementation of the integrator is a linked list of contact hypotheses indexed by the octree (See Fig. 4.13). Each leaf from the octree contains an iterator that points to an element of the list that contains all the hypotheses. The octree is used for nearest neighbor searches in the hypothesis space, meanwhile the list is used to access all the contact hypotheses sequentially (i.e. to check the hypotheses timestamp or threshold the hypotheses by its likelihood).

4.7 Experiments

To demonstrate and validate the sensor fusion framework proposed and show its suitability for the contact detection and localization problem, we have conducted two experiments: a validation and a use case.

On Tombatossals we have performed an experimental validation of the implemented contact hypotheses generators and the fusion method. On ARMAR-IIIb we have applied the contact detection method on a real grasping situation; using the contact output information to drive a reactive grasp algorithm like [Felip and Morales, 2009] or [Hsiao et al., 2010]. We have used the same voxel size in the generators and in the integrator, 5mm side. Thus, the precision of the contact detection is limited to the 5mm resolution

of the hypothesis space. The selected contact condensation method is the threshold, cluster and centroid.

4.7.1 Experimental validation

The experimental validation is performed using the Tombatossals robot. This experiment consists on touching three different objects (a box, a cylinder and a sprayer bottle) each one from 15 different approach directions for a total of 45 touch experiments.

For the evaluation two performance metrics are used:

- Detection rate: Measures the number of contacts detected over all the contacts that really occurred. This metric provides the likelihood that the used sensor modalities detect a contact.
- Localization error: To measure the precision, we compute the error of the detected contact location as the distance to the ground truth contact location. This metric provides information about the accuracy of the contact localization.

In order to validate the benefits of using a sensor fusion approach, the contact hypotheses provided by each generator, are recorded separately. Then, the contact detection results are obtained for each modality, without fusing data from different generators. After that, we obtain also the contact detection results using all the modalities at the same time. The results are evaluated with the proposed metrics.

Experimental setup

Environment

The scenario consists of an object on a table in front of the robot, the object is in a known position inside the arm workspace, so the robot can touch it easily from different approach directions.

Test objects

Three different objects are used for the validation experiments. A box, a cylinder and a sprayer bottle (See Fig. 4.14).

Assumptions

The 3D models of the objects are known. The position of the objects on the table is also known. For the execution of the experiments, the position is kept static in the real scenario. On the other hand, to simulate the uncertainty of state of the art object pose estimation algorithms (e.g. [Aldoma et al., 2012]), the pose of the object is modified with Gaussian noise in the simulator.

Figure 4.14: Test objects used for the validation of the contact detection framework. To show the scale of objects a 1 Euro coin is placed next to each object.

Ground truth data

To obtain the ground truth data, the position of the object is calibrated using the robot left arm. The calibration is manually performed: touching several points of the real object and moving the simulated object to fit those positions. With the object position calibrated in the simulator, we have used the joint positions recorded from the experiment execution to get the exact hand-object contact points and use them as ground truth data. Two of the experiments did not really touch the object, they were removed leaving 43 touches. To keep the ground truth valid, the object is fixed and cannot be moved by the robot during the experiments.

The role of the simulator in this experiment is twofold, as a ground truth tool and as a prediction engine.

Hypotheses generators

The hypotheses generators used for the validation experiments are: force-torque sensor, tactile sensors, range sensor and simulator.

In this case, the simulator is used as a prediction engine to generate support contact hypotheses. In order to model the uncertainty introduced by state of the art 3D object recognition and pose estimation methods [Aldoma et al., 2012] we have added a Gaussian error, $\mathcal{N}(\mu = 0, \sigma = 2)$ in cm, to the objects calibrated position.

Results and discussion

The results, after the execution of 43 touches, are shown in Table 4.1. The distance between the ground truth contact and the result of the contact condensation (See Sec.4.6.2) is used as the error measure ϵ. The standard deviation of the centroid calculation performed by the contact condensation is used as the precision measure σ.

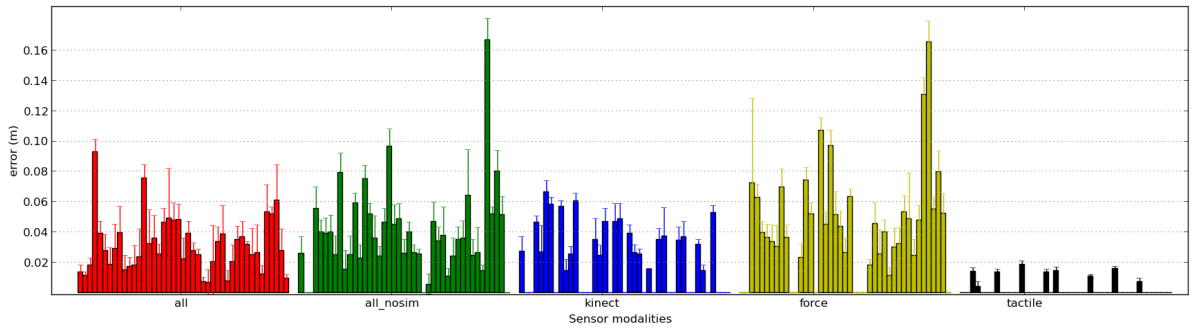


Figure 4.15: Results for the touch experiments considering different sensor modalities. Results are grouped by sensor modality, each one has 43 slots, one for each touch performed, if the contact was detected a bar shows the ϵ and σ for that contact.

Sensor Modality	Detected contacts	ϵ(cm)	σ(cm)
Tactile	9/43 (20.9%)	1.25	$\pm$0.20
Force	34/43 (79.1%)	5.37	$\pm$1.18
Range	25/43 (58.1%)	3.74	$\pm$0.73
All	39/43 (90.7%)	4.31	$\pm$1.10
All + Simulator	39/43 (90.7%)	3.33	$\pm$1.11

Table 4.1: Results for each sensor modality. Number and % of detected contacts. ϵ shows the distance between the ground truth and the detected contact. σ is the mean dispersion of the centroid calculation.

In Table 4.1 the mean accuracy and precision considering different sensor modalities is shown. Fig.4.15 depicts the individual results for each touch experiment considering different sensor modalities.

Although the tactile sensor modality has a small localization error (around 1cm ±0.2) the contact detection is quite low, 20.9%. This low detection rate is related to the reduced area that the tactile sensors can cover, thus many contacts happen outside the tactile sensor patches. Regarding the force-torque generator, although the detection rate is good (79.1%) the localization error is around (5cm ±1.18), the force-torque contact hypotheses generation method is very sensitive to noise and the effect of multiple contacts decreases the accuracy. The range modality shows average contact detection (58.1%) and good accuracy (3.7cm ±0.73), the main problem of this modality is that it requires the object to be moved in order to detect contacts.

Fusing the modalities, the detection raises to 90.7%. The accuracy depends on which sensors are detecting the contact. The fusion method automatically takes the most precise sensor (i.e. the hypotheses cloud with higher probability density). Note that the error (4.31cm ±1.1) is increased by those cases where only the force sensor generates hypotheses. This problem is solved adding the predicted contacts from the simulator, this reduces localization error by 23% leaving it at 3.33cm.

It is important to note that, beyond the sensors precision, the object position uncertainty (modelled by $\mathcal{N}(0, 2)$ cm), the robot model error and the joint encoders error also influence the final results.

4.7.2 Grasping application

To test the sensing framework on a real application, we have implemented a robust grasp primitive like the one presented in Sec.3.3.1 that uses the contact location output from the contact detection framework. The purpose is to test if the controller is able to grasp the bottle using the provided contact feedback.

Experimental setup

Environment

The scenario consists of a bottle on a table in front of the robot, the object is inside the arm workspace, so that the robot can grasp it without the need of moving the base.

Test objects

The test object consists on a transparent sparkling water bottle. The bottle is almost full as can be seen in Fig. 4.16.

Assumptions

The robot hand is placed approximately in front of the object, so it satisfies the initial condition for the grasp controller explained in Sec.3.3.1. There is no previous knowledge about the object shape, weight or other properties.

The object is not fixed and can be moved by the robot, only generators that depend on the object position (simulator predictions) would be influenced by this fact but for ARMAR-IIIb the simulator predictions are not implemented.

Hypotheses generators

The hypotheses generators used for the grasping experiments are: force-torque sensor, tactile sensors, finger pose feedback and motion estimation.

Results and discussion

The result of the experiment is shown in Fig. 4.16. The robot is able to detect the contact location fusing the information coming from the force sensor and from the arm motion. In this case, neither the tactile sensors nor the finger position did detect the contact. Using the detected contact location, the grasp controller can perform the corrective movements triggered by unexpected contacts and successfully grasp the target object even if the approach vector is not exactly in front of the object.

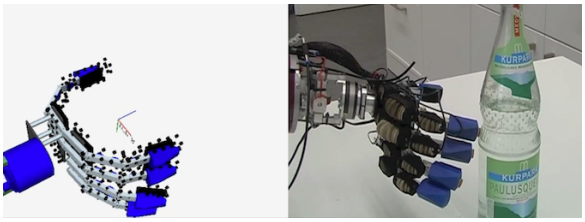
Under the experimental conditions, the sensor rate achieved was around 6Hz. It is important to note that all the system (sensor generators, integrator and visualization) were running on an average computer with two cores. As the hand was moving slowly for safety reasons, 6Hz were enough to react to the detected contact, correct the hand-object position and grasp the object successfully.

A more efficient implementation that decouples visualization from the contact detection framework, combined with a more efficient implementation of contact hypotheses generators resulted in a detection frame rate equal to the slowest contact hypotheses generator, faster than 30Hz.

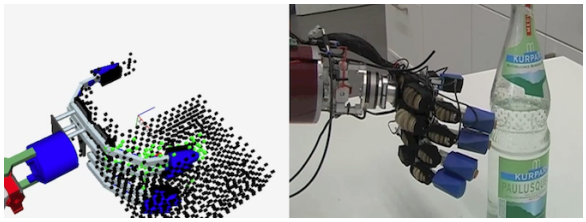
4.8 Conclusion

In this chapter we have shown theory, validation and application of a sensor fusion method focused on contact detection. This framework provides multi-modal contact event detection to our system, a key element in the proposed approach for human inspired robotic manipulation.

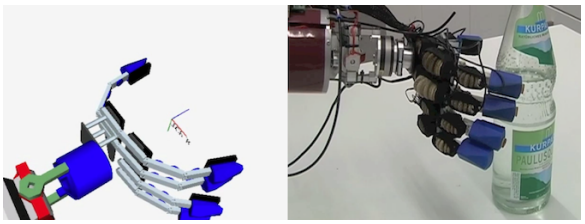
One of the contributions is that the method allows input from other sources but sensors, such as context, predictions or environment. We have shown that the projection of predictions or beliefs into the sensor space improves the results.



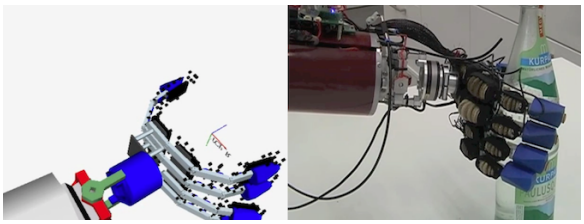
(a) Motion support hypotheses are generated while the hand moves.



(b) Contact detected. Force-torque hypotheses are fused with motion support hypothesis.



(c) Contact information is used to correct hand pose.



(d) Finally a good grasp is achieved.

Figure 4.16: Execution on a real robot. Black voxels: Discarded hypotheses. Green voxels: Hypotheses after likelihood threshold, Red voxels: Contact detection output.

The theoretical approach has been implemented in two different robotic platforms. The experiments carried out using Tombatossals have shown that the method is suitable to be used in real environments providing a framework to fuse sensor, simulation and prediction data to improve contact detection and localization. The experiments also show that the fusion of different sources performs better than the sources separated.

However, it is important to highlight that the likelihood threshold used in the contact condensation to discard low probability hypotheses is a key parameter. Its selection may change the behaviour of the sensor fusion system. Moreover, the clustering approach used to determine how many contacts are detected, has a computational cost very sensitive to the number of hypotheses considered. If the hypotheses number grows too high, it may affect the performance of the contact detection system.

Using a common representation for all the sensory inputs enables contact based controllers, presented in Chapter 3, to be more hardware independent. This makes systems more portable (same inputs), scalable (easy to add new sensors) and robust (failure tolerant). Moreover, as we show in this work, this level of abstraction enables the addition of non sensor data like: context, control or predictions. In this way we provide the manipulation primitive framework with a platform independent perceptual system that can be used to implement hardware agnostic primitives allowing the robots to share plans more easily.

The implementation on multiple platforms shows that the approach can contribute to the sensor skills of any robot or even enable the interaction of different robots on sharing and combining their knowledge about existing contacts. This opens the door to multi-robot scenarios, where contact hypotheses generators from different robots can be used together to share physical interaction information and detect contact events. The framework is not only limited to robots but to any source of contact information, it is also suitable for ambient intelligence.

The precision of the contact detection is limited by the accuracy of the contact hypotheses generators. Hence, further research on the proposed contact hypotheses generators or the addition of more precise sensors will help to improve the overall performance of the system. Moreover, with a few modifications the framework can also be used to detect surprise (prediction and sensing mismatch) and enable low level reactive behaviours, internal model refinement or higher level reasoning.

The described framework, contact generators and experiments were published in [Felip and Morales, 2014]. The RGBD based method used to implement the range sensor contact hypothesis generator was published in [Bernabe et al., 2013].