2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Congress Center Hamburg
Sept 28 - Oct 2, 2015. Hamburg, Germany

An Approach for a Distributed World Model with QoS-based Perception Algorithm Adaptation

Sebastian Blumenthal, Nico Hochgeschwender,
Erwin Prassler, Holger Voos and Herman Bruyninckx

Abstract— **This paper presents a distributed world model that is able to adapt to changes in the Quality of Service (QoS) of the communication layer by online reconfiguration of perception algorithms. The approach consists of (a) a mechanism for storage, exchange and processing of world model data and (b) a feedback loop that incorporates reasoning techniques to adapt to QoS changes immediately. The latter introduces a Level of Detail (LoD) metric based on a spatial resolution in order to infer an upper bound for the amount of data that can be transmitted without violating an application specific transmission delay.**

Experiments have been performed with Octree-based subsampling techniques applied to data originating from a RGB-D camera using simulated and real-world data sets for time-varying bandwidth values as employed QoS measure.

I. Introduction

With the advent of agile and versatile robot platforms, operating in the air, underwater and on the ground the development of Search and Rescue (SAR) missions [1] in unstructured and harsh environments such as alpine [2], maritime [3], and desaster [4] settings is becoming more and more a reality. In these application scenarios mixed teams of autonomous and semi-autonomous heterogenous agents such as humans, robots, and distributed sensors need to collaborate in order to achieve their tasks. To enable collaboration among these agents a world model needs to support physically distributed data storage and shared data access in order to improve the situational awareness of individual team members and the team as a whole. Here, a distributed world model creates and maintains a digital representation of the environment over a period of time based on the results of the employed perception algorithms. This representation needs to be exchanged by replication and synchronization among all agents. Recent approaches for distributed world modeling such as [5] and [6] do not tackle the requirements imposed by challenging SAR missions as described below.

To realize collaboration via distributed world models, a wireless networking and communication infrastructure is required. However, as Troubleyn *et al.* [7] pointed out real-world environments poses significant *Quality of Service (QoS)* challenges on the networking and communication infrastructure. More precisely, QoS *metrics* such as the time it takes to transmit a message from source to destination (delay) and the number of messages which are lost during transmission (message loss rate) are not necessarily known a-priori and can change over time. These time-varying QoS variations are caused by extreme environmental conditions such as heat and cold, agent mobility, and general application deployment in unaccessible environments such as disaster areas. The QoS properties often stand in contrast with the requirements coming from the application itself. For example, an operator in a SAR mission could formulate a max. tolerated message delay in order to receive certain information which is crucial to proceed with the mission.

To cope with these challenges in a systematic manner a representation of QoS properties and a mechanism to interpret them and eventually adapt the system behaviour is required. Some mechanism to adapt the system behaviour (a) reconfigure the QoS properties of the communication infrastructure [8] which is however framework dependent, or (b) reduce the amount of data that is sent by applying a data reduction [9] or compression [10] strategy which is challenging to adapt at run-time.

This work focuses on reducing the amount of data via appropriate adaptation of perception algorithms for data reduction at run-time. This includes an approach for a distributed world model with mechanisms for storage, exchange and processing of world model data (see Section II-A) and a feedback loop (see Section II-B) that incorporates reasoning techniques to adapt to QoS changes immediately. We introduce a Level of Detail (LoD) metric as a criteria to select an algorithm. We use bandwith as a QoS metric and and an according mapping to the LoD (see Section III). As illustrative example application we choose a setup involving a robot equipped with a Kinect RGB-D camera. This data has to be exchanged with one or multiple Human Machine Interfaces to be used by the rescuers. The experiments (see Section IV) show that the adaptation by algorithm selection satisfies an exemplary chosen application requirement of a max. transmission delay $t_{max\_delay} = 1\,\mathrm{s}$ even in the presence of changing QoS as faced in SAR missions. We make the following contributions:

- The extension of the Robot Scene Graph (RSG) [11] world model by a QoS-aware data exchange facility.
- The integration of Robot Perception Specification Language (RPSL) [12] to model existing data reduction algorithms in order to exploit its reasoning capabilities.
- The introduction of a novel LoD metric which is used to select an algorithm and a mapping to the bandwidth.

Sebastian Blumenthal and Herman Bruyninckx are with the Department of Mechanical Engineering, Katholieke Universiteit Leuven, Belgium. Nico Hochgeschwender and Erwin Prassler are with the Department of Computer Science, Bonn-Rhein-Sieg University, Sankt Augustin, Germany. Nico Hochgeschwender and Holger Voos are with the Automatic Control Laboratory, University of Luxembourg, Luxembourg. Corresponding author: sebastian.blumenthal@mech.kuleuven.be

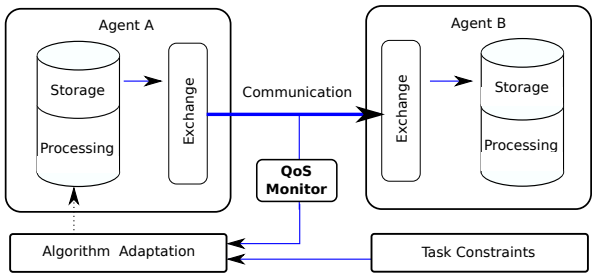

Fig. 1. Generic architecture for QoS aware data exchange between multiple agents. An agent embeds the mechanism for storage, exchange and processing of the data. A QoS Monitor enables the Algorithm Adaptation to reconfigure the processing modules of the agent.

II. Approach

A generalized overview of the presented approach is illustrated in Fig. 1. It consists of two major parts: (a) a mechanism for a data reduction work flow that can be reconfigured online and (b) a feedback loop that continuously measures the QoS and emits events to configure the reduction process.

The data reduction work flow is embedded into an agent. An agent comprises three modules. A **Storage** module that contains the data relevant for the target application. The data has to be modeled with a formal data model (e.g. a database scheme) because it is relevant for the reasoning module in the feedback loop. An **Exchange** module is responsible for encoding, decoding and transmitting data to other agents. The **Processing** facilities allow to apply any algorithm to the stored data and need to be reconfigurable at run-time.

The feedback loop consists of the **QoS Monitor** which reports on the current QoS of the communication layer. The **Algorithms Adaptation** module has a formal data model of the involved data types and the all available data reduction algorithms. This enables reasoning techniques to determine the most suitable algorithm given the current QoS situation. Application specific **Task Constraints** including preferences for trade-offs for the data representation have to be taken into account as well.

For the sake of readability the information flow in Fig. 1 is depicted in a unidirectional manner. The extension to a bidirectional composition means to apply the feedback loop to Agent B analogously.

A. Distributed World Model Mechanism

The generic architecture as described in the previous section can be applied to realize a distributed world model suitable for collaborative SAR missions as illustrated in Fig. 2. The agents **store** the world model data, thus we call them *World Model Agents*. The data-types have to comply to the data model of the *Robot Scene Graph (RSG)* [11] which has been introduced as a data representation for robotic world models shared across multiple agents. It essentially consists of different types of nodes that are organized in a graph structure. All nodes are addressable by Universally Unique IDs (UUID). An update is defined as the creation, deletion or modification of a single node and is propagated to an Update Monitor. One particular type of node that is relevant for the example application is the *GeometricNode* as it can store point cloud data. It is used to store the raw data of the 3D sensor of the robot.

The *Function Block* is a mechanism to perform **processing** within the *World Model Agent*. The input and output is specified via the UUIDs that describe a set of scene graph nodes. In this paper the Function Blocks are used to reduce the amount of data as they contain variations of *Octree*-based sub-sampling algorithms for point clouds. Here the used policy is to trigger the blocks on every insertion of sensor raw data.

The employed **Update Monitor** controls the propagation of changes by a LoD metric (see Section III). For every update the LoD value of the given point cloud data is computed and it is only forwarded if it is lower than or equal to the max. allowed LoD value. This mechanism prevents sending data that is not suitable to be transmitted given the current QoS condition of the communication framework.

The distributed world model uses HDF5 [13] to **encode** and **decode** the update messages. HDF5 is a file format for storing large scale scientific datasets. The datasets are hierarchically composable. Here the HDF5 file format is used, mainly because it shares similarities with the RSG data model which eases the mapping of the RSG data types to HDF5 structures. Instead of defining our own data protocol we rather map the RSG data model to the HDF5 data model and obtain a serialization facility without further implementation efforts. The fact that a framework independent serialization is chosen allows to make statements on the size of the transmitted data as seen later for point cloud data (see Section IV-B).

Conceptually the connectivity between the World Model Agents is a *peer-to-peer* topology. A **Bridge** sends its data via the communication framework to the bridge of another agent.

The role of the **Configurator** is to store *which* Function Block has to be executed when new raw data arrives from the sensor. It also stores the maximum allowed LoD value that is used within the Update Monitor.

The distributed world model provides the infrastructure for storage, exchange and processing of world model data. It does not know *why* a particular algorithm i.e. Function Block is selected for which reason. This aspect is completely delegated to the **Perception Algorithm Adaptation** module as explained in the next section.

B. Perception Algorithm Adaptation

The Robot Perception Specification Language (RPSL) introduced in [12] is used to represent the knowledge required for the data reduction algorithm selection. It is a Domain-Specific Language (DSL) which enables to model two crucial elements of perception systems in a declarative and formal manner, namely *perception graphs* and the *data types*. In RPSL a Perception Graph (PG) is a composition of components in the form of a Directed Acyclic Graph

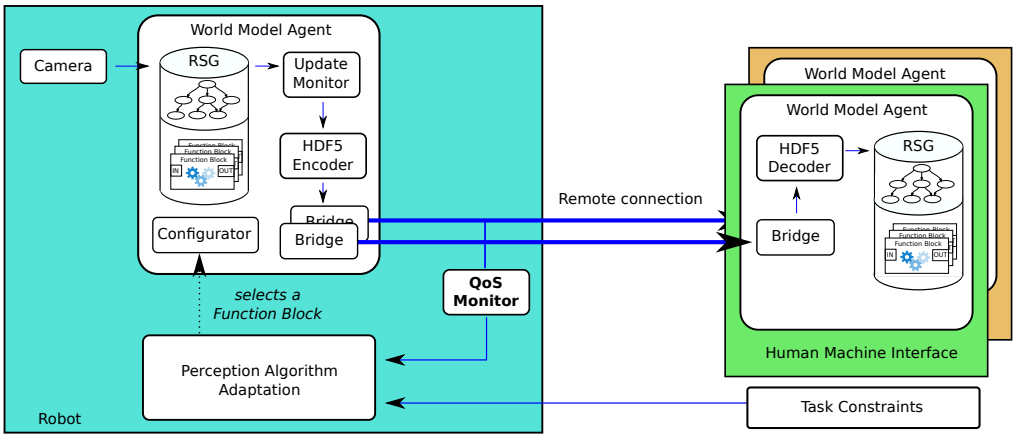

Fig. 2. Architecture for a QoS aware data exchange between multiple World Model Agents applied to the distributed world model for a collaborative SAR mission involving a robot and one or multiple HMIs. The data model, its processing and encoding/decoding complies to the RSG approach. The feedback loop is realized with RPSL in order to select algorithms from the perception domain.

(DAG). Here, components are distinguished between (a) sensor components representing sensors such as cameras and (b) processing components implementing perceptual algorithms such as data reduction methods. In the context of this work a PG is used to model *what* precisely a single Function Block in the Robot Scene Graph does and encodes one particular data reduction algorithm. As PGs consume and produce data on spanning multiple levels of abstractions ranging from raw sensor data and subsymbolic representations to symbolic information a suitable knowledge representation mechanism is required. In RPSL the vector-based Conceptual Space (CS) [14] knowledge representation framework is employed. A CS contains the following constituent parts:

- A **Conceptual Space** is a metric space where **Concepts** are defined as convex regions in a set of domains (e.g. the concept *Level of Detail*).
- A **Domain** includes a set of **Domain Dimensions** that form a unit and are measurable (e.g. a single dimension for the LoD).
- An **Instance** is a specific vector in a space (e.g. a LoD value of 1.0 $\frac{sample}{m^3}$).
- A **Prototype** is an **Instance** which encodes typical values for a Concept.

In the context of this work the CS representation is used to model the data in- and output of perception graphs (e.g. concepts of different point clouds with and without color) and their corresponding LoD values in the form of prototypes (see LoD_{max} in Table II). Hence, each stored algorithm is attached with a LoD prototype which is later used to compute the most suitable algorithm. To select an appropriate data reduction algorithm a QoS value is first turned into a LoD prototype which is later compared to all other existing LoD prototypes stored in the repository. In this paper we apply the Euclidean distance as a simple, yet powerful comparison metric. An algorithm is chosen based on the smallest distance between the requested and the stored prototypes. The experimental section solely exploits the LoD concept whereas further concepts expressed by the task constraints such as existence or absence of color information in a point cloud can be incorporated. This yields in an extension of the search space for an appropriate data reduction algorithm.

To be able to account for QoS changes a mapping from measured bandwidth to the LoD is required.

III. Mapping of LoD to Measured Bandwidth

We introduce the Level of Detail (LoD) as a generic metric to define a spatial resolution of point clouds independent of the actual data representation (i.e. no byte consumption is directly inferable). It is used to (a) describe a spatial resolution of a given point cloud and (b) to describe the boundaries of the output of a data reduction (in our case sub-sampling) algorithm. We define LoD as follows

$$LoD = \frac{N_{points}}{V}, \quad \text{in } \frac{sample}{m^3} \tag{1}$$

where N_{points} is the number of point samples per volume V. This formula is somewhat similar to printer specifications that use *dots per inch*.

The proposed architecture (see Fig. 2) measures in the domain of *bandwidth* which is not directly connected to the representation used in the perception domain (LoD). The intention is the keep the domains for communication and perception separated according to the *separation of concerns* principle (otherwise the LoD is not reusable in other applications). Therefore, we define a formal mapping between both domains.

For the mapping between LoD and bandwidth we define f as follows $f : B \rightarrow LoD$ where B denotes the domain of bandwidth in bytes per seconds and LoD as defined in Eq. 1. To compute the LoD we define the following parameters which can be derived for an application a-priori.

- t_{max_delay}: is the max. delay in seconds tolerated by the application.
- V_{max_space}: is the max. volume covered by the sensor. It can be deduced from the specification of the sensor.

TABLE I
COSTS OF ENCODING AND DECODING POINT CLOUDS WITH HDF5.

N_{points}	t_{encode}	t_{decode}	$bytes$	$bytes_{per\_point}$
10	1.919 ms	20.621 ms	11584	$\approx$ 1158
100	4.130 ms	36.932 ms	43616	$\approx$ 436
1000	5.690 ms	38.541 ms	32517	$\approx$ 32
10000	7.659 ms	40.452 ms	331616	$\approx$ 33
100000	9.363 ms	44.643 ms	3211616	$\approx$ **32**

- $bytes_{per\_point}$: is the relation between the number of points in a point cloud and it's corresponding message size in bytes.
- t_{offset}: is the worst case total time required for encoding and decoding a message.

As the bandwidth $b \in B$ is a variable that varies at run-time we derive the maximum time required to send a message as $t_{max} = t_{max\_delay} - t_{offset}$. The number of max. bytes that can be send depends on the available bandwidth: $bytes_{max} = t_{max} * b$. As we know the memory consumption of a point in a point cloud we can estimate the max. number of points to be sent in a message: $N_{points} = bytes_{max}/bytes_{per\_point}$.

From the V_{max_space} max. covered volume and the max. allowed points we can deduce a max. LoD: $LoD_{max} = N_{points}/V_{max\_space}$. This leads to mapping f:

$$LoD_{max} = f(b) = \left(\frac{(t_{max} * b)/bytes_{per\_point})}{V_{max\_space}} \right) \quad (2)$$

.

This LoD_{max} denotes an upper bound for the LoD to satisfy the application tolerance t_{max_delay}. It is used to formulate a request to the **Perception Algorithm Adaptation** module in the presence of bandwidth changes. In the experimental section algorithms are selected that are capable of reducing point cloud data up to a max. LoD value.

IV. EXPERIMENTAL VALIDATION

A. Experimental Objectives and Hypotheses

The intention is to assess the feasibility of the approach i.e. to satisfy an application based tolerance on the max. delay while having a variable bandwidth of the communication layer. An assumption for the experiments is that the latency of the connection itself remains stable. We have the following main objectives and hypotheses:

- **Objective 1**: The tolerated max. delay which is defined by the application always holds even in the presence of changing QoS i.e. bandwidth.
- **Objective 2**: The lower the available bandwidth, the lower the number of points that are transmitted.
- **Hypothesis 1**: The density of the Perception Graph repository affects the delay significantly.

To assess the objectives and hypothesis above we also formulate the following side hypotheses:

- **Hypothesis 2**: The consumed time for encoding and decoding of messages has each a predictable relation to the number of points.
- **Hypothesis 3**: The message length has a predictable relation to the number of points.

B. Experimental Design

1) Parameters for the LoD to Bandwidth Mapping: First, for the experiments the set of parameters required for the LoD to bandwidth mapping (see Section III) has been experimentally derived: artificial point cloud data sets have been fed to the system with an increasing number of points. The according durations for t_{encode} for encoding and t_{decode} for decoding and the resulting message size $bytes$ are listed in Table I.

In our case the HDF5 encoding and decoding reveals a predictable monotonic increasing characteristics. This *validates* our side **Hypothesis 2**. Though, for the experiments we choose $t_{offset} = 0\,\mathrm{s}$. A validation of **Objective 1** with this value is still valid for more realistic values for t_{offset}.

We approximate the relation $bytes_{per\_point} = 32$ as it converges to this value for large point clouds. The side **Hypothesis 3** is considered to be *true*. Furthermore, as we are using a Kinect RGB-D sensor we conservatively approximate $V_{max\_space} = 64\,\mathrm{m}^3$ accounting for the bounding box[1] dimensions of $4\,\mathrm{m} * 4\,\mathrm{m} * 4\,\mathrm{m}$. The max. tolerable delay for the example application is set to $t_{max\_delay} = 1\,\mathrm{s}$ (see Section I).

2) Algorithm Selection for the Perception Graph Repository: *Octree*-based sub-sampling [15] filtering is a commonly used strategy to reduce point cloud data. Here we choose a set of *Octree* filters with different leaf sizes.

We recapitulate the relation between *Octree* leaf size N_{leaf} and max. possible LoD, given the parameters from the above section: The smaller the leaf size, the higher the possible resolution. A leaf size $N_{leaf} = 1\,\mathrm{m}$ means all points (if any) in a leaf with size $1\,\mathrm{m} * 1\,\mathrm{m} * 1\,\mathrm{m}$ are discarded and represented by the center point of that cube. One point per such unit cube is exactly $LoD_{max} = 1\ \frac{sample}{m^3}$. The according formula for other leaf sizes is: $LoD_{max} = 1/(N_{leaf})^3$

The selection focuses on a set that emphasizes more variability within a bandwidth range for cell phone networks i.e. a bandwidth range of $\approx 10^5$ - 10^6 bytes per second that corresponds according to equation (2) to a LoD range from $97.66\ \frac{sample}{m^3}$ - $976.56\ \frac{sample}{m^3}$. Table II repesents the chosen algorithms including all leaf size and corresponding LoD values.

A dummy algorithm called `camera` is also stored in the repository to account for situations where enough bandwidth is available. In this case the raw data is not changed at all.

3) Experiments for the main objectives: The following two experiments have been designed. Both use bandwidth as *controlled variable*, the parameters from Section IV-B.1 and the PG repository as indicated in Section IV-B.2 . As point cloud dataset for the camera an office scene is chosen.

- **Experiment A:** Uses a full bandwidth spectrum with respect to existing communication technologies. I.e. the values start at zero, are incremented each after $10\,\mathrm{s}$,

[1]These values can be enforced with an appropriate pre-filtering step.

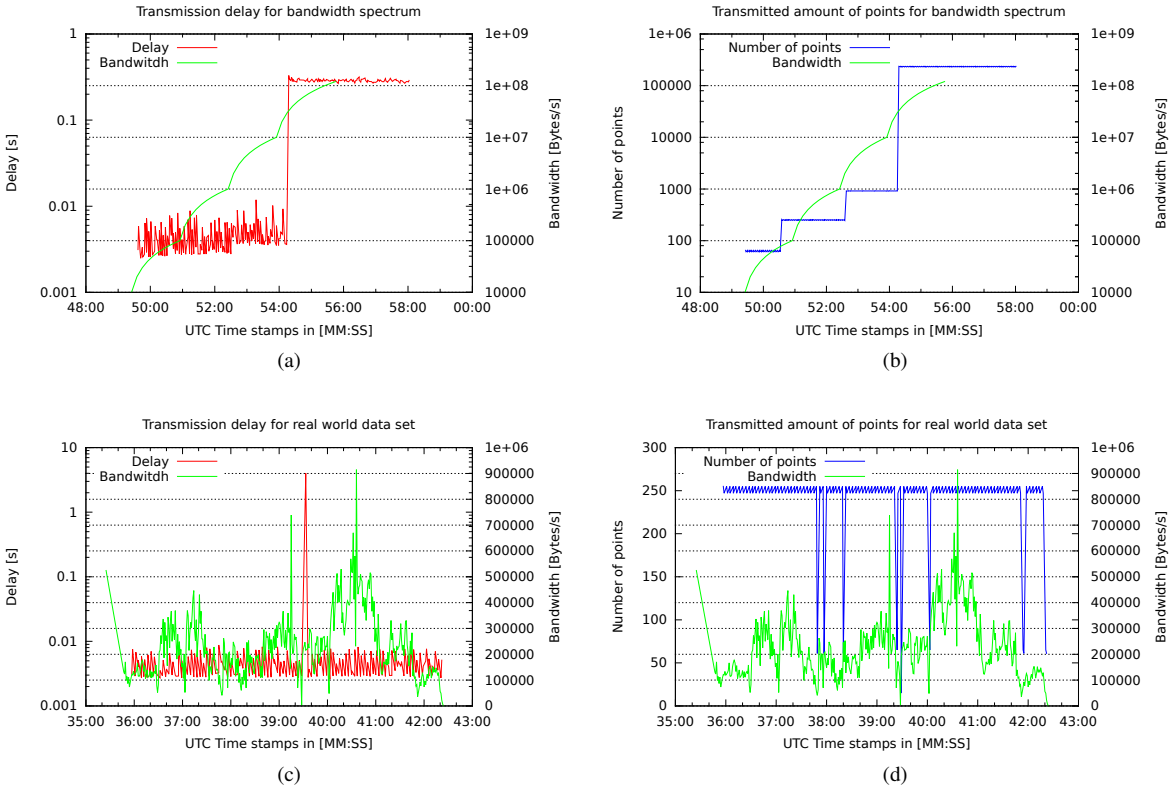

Fig. 3. Experimental results. Plot (a) presents the measured transmission delay while (b) indicates the transmitted number of points for the full bandwidth spectrum data set. Accordingly, (c) and (d) show the used real-world data set.

TABLE II
USED PERCEPTION GRAPH REPOSITORY.

Name	N_{leaf} in $[m]$	LoD_{max} in $[\frac{sample}{m^3}]$
`octreefilter1`	1	1
`octreefilter0.5`	0.5	8
`octreefilter0.25`	0.25	64
`octreefilter0.2`	0.2	125
`octreefilter0.1`	0.1	1000
`camera`	-	10000

and stop at 10^8 bytes per seconds. The range includes typical bandwidths for cell phone networks (10^5 - 10^6), WiFi (10^6 - 10^7) and Ethernet (10^7 - 10^8).

- **Experiment B:** Some SAR missions propose to use cell phone networks [3] as one part of their communication infrastructure. Hence, a real-world data set called *Car Snaroya Smestad*[2] [16] is chosen to test the system under realistic bandwidth settings for such cell phone networks.

C. Experiment Execution

All experiments have been performed on a single off-the-shelf Laptop with Intel Core i7 CPU and ≈ 8GiB RAM and Ubuntu 12.04. The used C++ compiler is g++ version 4.6.4 and the used Ruby interpreter has version 1.9.1.

[2]File name: report.2011-02-14_2032CET

The BRICS_3D[3] implementation of the RSG is used together with a set of *Octree*-based sub-sampling Function Blocks[4]. For RPSL we use the Ruby based implementation[5].

As communication framework ROS Hydro is used. Two World Model Agents are each embedded into a ROS node. To be able to control the bandwidth an additional relay node is employed that introduces an artificial delay to the used topic based on a configurable bandwidth parameter. A bandwidth generator node sends simultaneously the bandwidth parameter according to the experimental design to the relay node and the perception adaptation

D. Experiment Results and Analysis

For **Experiment A** (see Fig. 3a and Fig. 3b) the delay tolerated by the application always holds with a maximum delay of ≈ 0.3 s. As shown in Fig. 3b the adaptation decision which leads to increased or decreased number of points is clearly visible in discrete steps. This verifies **Objective 2**. Each discrete step represents a different selection of an algorithm. The last step represents the transmission of the raw camera data as the bandwidth becomes sufficient. This is caused by the selection of the `camera` PG. The granularity of discrete steps is dense in the cell phone bandwidth range whereas in the WiFi range the selection is rather sparse.

[3]`github.com/brics/brics_3d`
[4]`github.com/blumenthal/brics_3d_function_blocks`
[5]`github.com/nicoh/RPSL`

This can be explained with the chosen algorithms and corresponding parameters (see Table II) stored in the repository. We argue that an application developer needs to consider this effect at design-time through a carefully population of the PG repository. By doing so the application developer can influence the overall performance of the adaptation. In our case we emphasized the cell phone bandwidth range. **Hypothesis 1** seems to be correct.

For **Experiment B** (see Fig. 3c and Fig. 3d) which uses a real-world data set in the cell phone bandwidth range the delay tolerated by the application holds mostly with one exception. This outlier occurs on a sudden change to a very low bandwidth value (2736). In this case the system selected `octreefilter1`, however the reduction was not enough. This input value is beyond the range of the what algorithms in the repository are capable of. The plot indicated in Fig. 3d shows that mostly a rather similar data reduction has been applied. The selection toggles between `octreefilter0.2` and `octreefilter0.25` but as the leaf cell parameters are so similar to each other they produce nearly the same output. At the exceptional case that misses the application tolerance 15 points have been transmitted. 2 points would have still met the requirement. This shows a limitation of the approach: it only performs well if the repository has been carefully designed to cope with all possible cases. Also the extreme case for $b = 0$ cannot be handled with this setup. A policy for the Update Monitor to not send anything below a certain threshold could be a remedy for that.

Both experiments support **Objective 1**. The approach is able to handle QoS changes for this experimental setup.

V. Conclusion and Future Work

This paper has presented an approach for a generic QoS aware data exchange between multiple agents. It is applied to an example application for a SAR mission that has to adapt to QoS changes immediately. The approach realizes a mechanism for storage, exchange and processing of world model data by employing the data model and processing facilities of the Robot Scene Graph (RSG). The feed backloop utilizes the Robot Perception Specification Language (RPSL).

We clearly separated world modeling (the *what*), perception (the *how*) and QoS communication aspects. For the latter two domains we introduced a mapping of the LoD parameter to the used QoS metric: bandwidth.

The proposed approach allows to have a dedicated data exchange mechanism while the Perception Graph (PG) repository can be filled with any algorithm or algorithm combination. Though, we only selected a single type of algorithm, namely an *Octree*-based sub-sampling technique to prove the overall feasibility. This has been experimental validated for the used example setup. The experiments reveal that the population density of the repository has a strong effect on the granularity of the adaptation and the boundary case for low bandwidth.

Extension of the repository with more diverse algorithms, other QoS concepts, integration on concrete robot platforms and incorporation of task constraints in the experimental setup are promising directions for future work. This will also help to study the connection between the tolerable transmission delay, required LoD, and the overall performance of the task to achieve.

ACKNOWLEDGMENTS

The authors acknowledge the support from the KU Leuven Geconcerteerde Onderzoeks-Acties *Model based intelligent robot systems* and *Global real-time optimal control of autonomous robots and mechatronic systems*, and from the European Union's 7th Framework Programme (FP7/2007–2013) projects *BRICS* (FP7-231940), *ROSETTA* (FP7-230902), *RoboHow.Cog* (FP7-288533), and *SHERPA* (FP7-600958). Further, Nico Hochgeschwender received a PhD scholarship from the Graduate Institute of the Bonn-Rhein-Sieg University which is gratefully acknowledged.

References

[1] R. Murphy, S. Tadokoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, and A. Erkmen, "Search and rescue robotics," in *Springer Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. Springer Berlin Heidelberg, 2008, pp. 1151–1173.

[2] L. Marconi, S. Leutenegger, S. Lynen, M. Burri, R. Naldi, and C. Melchiorri, "Ground and aerial robots as an aid to alpine search and rescue: Initial sherpa outcomes," in *Safety, Security, and Rescue Robotics (SSRR), 2013 IEEE International Symposium on*, Oct 2013, pp. 1–2.

[3] G. De Cubber, D. Doroftei, D. Serrano, K. Chintamani, R. Sabino, and S. Ourevitch, "The eu-icarus project: Developing assistive robotic tools for search and rescue operations," in *Safety, Security, and Rescue Robotics (SSRR), 2013 IEEE International Symposium on*, Oct 2013, pp. 1–4.

[4] R. Sheh, T. Kimura, E. Mihankhah, J. Pellenz, S. Schwertfeger, and J. Suthakorn, "The robocuprescue robot league: Guiding robots towards fieldable capabilities," in *ARSO'11*, 2011, pp. 31–34.

[5] M. Tenorth, A. C. Perzylo, R. Lafrenz, and M. Beetz, "Representation and exchange of knowledge about actions, objects, and environments in the roboearth framework," *IEEE Transactions on Automation Science and Engineering (T-ASE)*, vol. 10, no. 3, pp. 643–651, 2013.

[6] A. Dietrich, S. Zug, S. Mohammad, and J. Keiser, "Distributed management and representation of data and context in robotic applications," in *Proceedings of the IEEE/RSI International Conference on Intelligent Robobts and Systems (IROS)*, Chicago, Illinois, 2014.

[7] E. Troubleyn, I. Moerman, and P. Demeester, "QoS Challenges in Wireless Sensor Networked Robotics," *Wireless Personal Communications*, vol. 70, no. 3, pp. 1059–1075, 2013.

[8] M. Eich, R. Hartanto, S. Kasperski, S. Natarajan, and J. Wollenberg, "Towards coordinated multirobot missions for lunar sample collection in an unknown environment," *Journal of Field Robotics*, vol. 31, no. 1, pp. 35–74, 2014.

[9] J. Kammerl, N. Blodow, R. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, May 2012, pp. 778–785.

[10] A. Birk, S. Schwertfeger, and K. Pathak, "A networking framework for teleoperation in safety, security, and rescue robotics," *Wireless Communications, IEEE*, vol. 16, no. 1, pp. 6–13, 2009.

[11] S. Blumenthal, H. Bruyninckx, W. Nowak, and E. Prassler, "A Scene Graph Based Shared 3D World Model for Robotic Applications," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany*, 2013.

[12] N. Hochgeschwender, S. Schneider, H. Voos, and G. Kraetzschmar, "Declarative specification of robot perception architectures," in *Proceedings of the International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, Bergamo, Italy, 2014.

[13] M. Folk, G. Heber, Q. Koziol, E. Pourmal, and D. Robinson, "An overview of the HDF5 technology suite and its applications," in *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*. ACM, 2011, pp. 36–47.

[14] P. Gärdenfors, *Conceptual spaces: The geometry of thought*. The MIT Press, 2004.

[15] A. Nüchter, *3D Robotic Mapping The Simultaneous Localization and Mapping Problem with Six Degrees of Freedom*. Springer, 2009.

[16] H. Riiser, P. Vigmostad, C. Griwodz, and P. Halvorsen, "Commute path bandwidth traces from 3g networks: analysis and applications," in *Proceedings of the 4th ACM Multimedia Systems Conference*. ACM, 2013, pp. 114–118.