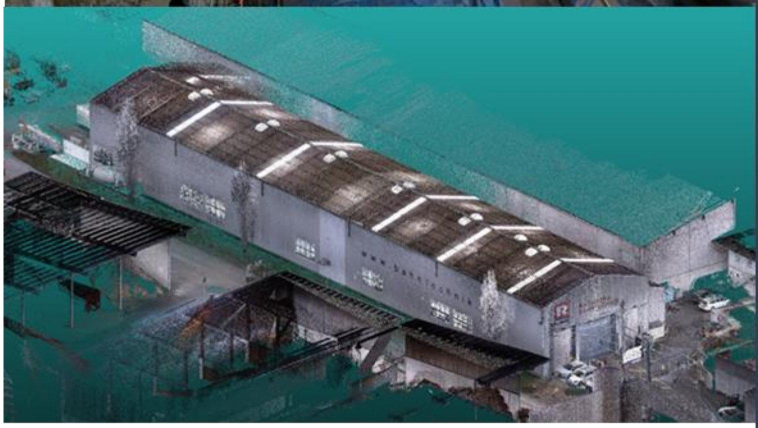


COGITO

CONSTRUCTION PHASE
DIGITAL TWIN MODEL

cogito-project.eu

D5.8 – User interface and AR enabled In-situ QC Visualization v2

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 958310

D5.8 – User interface and AR enabled In-situ QC Visualisation v2

Dissemination Level:	Public
Deliverable Type:	Demonstrator
Lead Partner:	CERTH
Contributing Partners:	Hypertech, UEDIN, FER, OLOD, RSRG
Due date:	31-10-2022
Actual submission date:	27-10-2022

Authors

Name	Beneficiary	Email
Thanos Tsakiris	CERTH	atsakir@iti.gr
Evangelia Pantraki	CERTH	epantrak@iti.gr
Apostolia Gounaridou	CERTH	agounaridou@iti.gr
Michalis Chatzakis	CERTH	mchatzak@iti.gr
Tasos Sinanis	CERTH	sinanis@iti.gr
Vasilis Dimitriadis	CERTH	vdimitriadis@iti.gr
Pavlos Kakaratzas	Hypertech	p.kakaratzas@hypertech.gr
Panagiotis Tsatsoulis	Hypertech	p.tsatsoulis@hypertech.gr

Reviewers

Name	Beneficiary	Email
Giorgos Lilis	UCL	g.lilis@ucl.ac.uk
Jochen Teizer	DTU	teizerj@dtu.dk

Version History

Version	Editors	Date	Comment
0.1	CERTH	30.08.2022	Table of Contents
0.4	CERTH	19.10.2022	Submission for peer review
0.5	UCL, DTU	21.10.2022	First draft internal review
0.7	CERTH	24.10.2022	Version updated after reviews
0.8	CERTH, Hypertech	26.10.2022	Final checks
0.9	CERTH	27.10.2022	Final version
1.0	CERTH, Hypertech	27.10.2022	Submission to the EC portal

Disclaimer

Executive Summary

The COGITO Deliverable D5.8 “User interface and Augmented Reality (AR) enabled In-situ QC Visualisation v2” documents the second version of the COGITO Digital Twin visualisation with Augmented Reality (DigiTAR) module and focuses on the development activities concerning the T5.4 “User Interface for Construction Quality Control”. Overall, DigiTAR operates as a Graphical User Interface (GUI) for visualising Construction Safety (CS) and Quality Control (QC) results as well as for on-site visual data capturing and pre-processing.

The DigiTAR module has three main functionality modes: the QC mode, the Safety mode, and the Pre-processing mode. Within the fore mode, the user can visualise the QC results provided by the Geometric and Visual QC COGITO components in situ in order to confirm the automatically detected defects and propose the necessary remedial works. In addition, by selecting the Safety mode, the potentially hazardous areas generated by the Safety-related COGITO components can be displayed on-site in order to confirm them and propose additional mitigations. Furthermore, DigiTAR could be considered as the GUI of the Visual Data Pre-processing module; hence, the user is able to create a new pre-processing job and add the related information (e.g., filter parameters, capture device properties). Finally, since DigiTAR runs on HoloLens, it is considered as a data acquisition tool, which captures visual data on-site in order to forward it for pre-processing and further Quality Control. The DigiTAR module overall architecture is analysed in this deliverable. The tool is composed of three main layers: the main Application layer, the Viewers’ layer, and the Communication layer. The first oversees implementing the main functionalities of the tool, while the second one is responsible for visualising 2D and 3D data on-site; the last one manages the communication with other COGITO components.

The present documentation of the COGITO User interface and AR enabled In-situ QC Visualisation, along with its sub-components, is oriented towards the functionalities they broadly deliver, the technology stacks they build upon, the inputs, outputs, and APIs they expose, the installation instructions, the assumptions and restrictions, the usage walkthrough, the development and integration status, and the requirements coverage. In this second release, refinements are implemented with regards to the QC and the Pre-processing modes, while additional functionalities are implemented with regards to the Safety mode.

Table of contents

List of Figures

List of Tables

List of Acronyms

Term	Description
AR	Augmented Reality
COGITO	Construction Phase diGItal Twin mOdel
DigiTAR	Digital Twin Visualisation with Augmented Reality
DT	Digital Twin
GUI	Graphical User Interface
H&S	Health and Safety
IFC	Industry Foundation Classes
QC	Quality Control
CS	Construction Safety

1 Introduction

1.1 Scope and Objectives of the Deliverable

This deliverable reports on the work conducted from M20 to M24 on the User Interface for Construction Quality Control that is developed as part of task T5.4. The scope of the Digital Twin visualisation with Augmented Reality component (DigiTAR) is to visualise the results of the Quality Control (QC) components (Geometric and Visual) as well as potential construction site hazards (Health and Safety issues, H&S) in-situ. In addition, DigiTAR is in charge of collecting as-built data and implementing part of the visual data pre-processing on-site. Subsequently, DigiTAR operates as a Graphical User Interface (GUI) for both construction Quality Control and on-site visual data pre-processing: (a) The stakeholders can view and confirm the results generated by the QC and Safety components in-situ as well as propose remedial works and mitigations for them respectively, (b) they can capture image data on-site, assign them to specific jobs and IFC components, prepare and display them after pre-processing; the data are eventually sent for Visual QC.

More specifically, this deliverable reports on the development of the second release of the DigiTAR component focusing on its main layers listed below:

- **Viewers Layer:** is in charge of visualising the necessary data (2D images and 3D IFC models).
- **Application Layer:** includes the basic functionalities of the application such as data capture, mode selection, job, and report generation.
- **Communication Layer:** manages the communication and the data exchange between the DigiTAR component and other COGITO components (namely the Digital Twin (DT) Platform and the Visual Data Pre-processing module).

1.2 Relation to other Tasks and Deliverables

T5.4 "User Interface for Construction Quality Control" and consequently D5.8 "User interface and AR enabled In-situ QC Visualisation v2" are related to the following COGITO tasks and deliverables:

- The second version of the COGITO architecture in the corresponding deliverable "D2.5 COGITO System Architecture v2" provided an overview on the DigiTAR module, its requirements, and the communication with other components.
- The DigiTAR module, similarly to all components, relies on a shared ontology and a common data model developed within T3.2 "COGITO Data Model, Ontology Definition and Interoperability Design"; the second version of COGITO ontologies and data models have been documented in D3.3 "COGITO Data Model, Ontology Definition and Interoperability Design v2".
- The DigiTAR module uses as input the results generated by the Geometric QC component to visualise and confirm on-site the detected geometric defects and their location (D5.5 and D5.6 "BIM-based Standard Test Methods for Geometric Quality Control" v1 and v2, respectively).
- The DigiTAR module uses as input the results generated by the Visual QC component to visualise and confirm on-site the detected visual defects and their location (D5.3 and D5.4 "Deep-Learning - based Visual QC component" v1 and v2, respectively).
- The DigiTAR module uses as input the results generated by the SafeConAI tool to visualise and confirm on-site the identified hazardous regions (D4.1 and D4.2 "Preventive Health & Safety Application" v1 and v2, respectively).
- The DigiTAR module communicates with the Visual Data Pre-processing module to apply filters in 2D images captured by Microsoft HoloLens on-site, preview and finalise the processed data that is finally sent for Visual Quality Control (D3.7 and D3.8 "Visual Data Pre-processing Module" v1 and v2, respectively).

1.3 Structure of the Deliverable

Remaining sections of this deliverable are organised as follows:

- Sub-section 2.1 presents the overall architecture of the DigiTAR module, introducing its sub-components and its workflow diagrams.
- Sub-section 2.2 describes the technologies, libraries and tools utilised for the implementation of this specific module.
- Sub-section 2.3 notes the inputs and outputs of the component as well as the APIs documentation.
- Sub-section 2.4 provides a brief manual (guidelines) on how to use this specific component.
- Sub-section 2.5 refers to information about the source code repository, the delivery form, and the license of the component.
- Sub-section 2.6 describes how the DigiTAR module is accessible by the user.
- Sub-section 2.7 presents the status of the component and provides a plan regarding the integration procedure.
- Sub-section 2.8 notes the functional, non-functional and stakeholder requirements that are covered by the component.
- Sub-section 2.9 describes the assumptions already made for this component, as well as restrictions and measures that will be taken into consideration for the COGITO System Integration.
- The document concludes with Section 3, where the progress, the next steps and the contribution to the overall COGITO objectives are being reported.

1.4 Updates to the First Version of the DigiTAR Module

Since the submission of D5.7 in M19, the COGITO User interface and AR enabled In-situ QC Visualisation Module has evolved as in the following:

- **Refinements to the 3D IFC viewer sub-module.** The 3D BIM model visualisation was (i) optimized in terms of visual appearance to provide a more user-friendly interface and (ii) extended by implementing and providing a variety of visualisation methods, which are presented in detail in Section 2.4.2.
- **Updates and refinements to the communication with the Visual Data Pre-processing module**. The bilateral communication between the DigiTAR module and the Visual Data Pre-processing module has been refined and simplified in this second and final version of the tool.
- **Updates and refinements to the visualisation of the Quality Control results.** In this version of the DigiTAR tool, in the Quality Control mode, 3D tags are pinned on the 3D elements that are included in the QC results. Both Geometric and Visual QC results are visualised within this version of the DigiTAR tool.
- **Updates and refinements to the visualisation of elements to be included in the Pre-processing jobs.** In this version of the DigiTAR tool, in the Pre-processing mode, the 3D IFC elements that are included in the Visual Quality Control workorders are highlighted in red to notify the user. Subsequently, the user can generate pre-processing jobs and attach these elements to them.
- **Updates and refinements to the 3D IFC viewer sub-module for the safety enriched IFC file**. The updated version of the enriched with safety information IFC file is visualised within this version of the DigiTAR tool. The 3D IFC viewer sub-module was updated to parse the safety properties (documented in D4.2 “Preventive Health & Safety Application” v2) that were injected into the IFC file.
- **Implementation of the user authentication and authorization**. In this version of the tool, the authentication and authorization of the users is implemented through Keycloak.
- **Preparative implementation of DT Platform integration.** In this version of the tool, communication with the DT Platform has already been established for retrieving: (i) the project list, (ii) the BIM model for the selected project, and (iii) the QC results. During the integration process (T8.1 End-to-end ICT System Integration, Testing and Refinement), the communication between the DigiTAR (as well as the rest COGITO components) and the DT Platform will be extended, tested, and finalised.

2 Digital Twin Visualisation with Augmented Reality (DigiTAR)

In this section, the DigiTAR module is presented in detail. In Section 2.1, the overall architecture of the component as well as the different sub-components it is comprised of is presented. Information about the implementation tools is presented in Section 2.2. The input and output data of the DigiTAR module are presented in Section 2.3, along with the API documentation. A detailed usage walkthrough is provided in Section 2.4. Information regarding the licensing of the DigiTAR module is presented in Section 2.5, while installation instructions are provided in Section 2.6. The development and integration status of the tool is discussed in Section 2.7. Finally, Sections 2.8 and 2.9 present the requirements coverage for this version of the DigiTAR module and some basic assumptions and restrictions, respectively.

2.1 Overall Architecture of the DigiTAR Module

The DigiTAR module has three separate modes: (i) the Quality Control mode, (ii) the Safety mode and (iii) the Visual Data Pre-processing mode. The first one is in charge of visualising the QC (Geometric and Visual) results on-site in order to be confirmed by the relevant stakeholders. Based on their decision, additional remedial works can be assigned to specific IFC components (Figure 1).

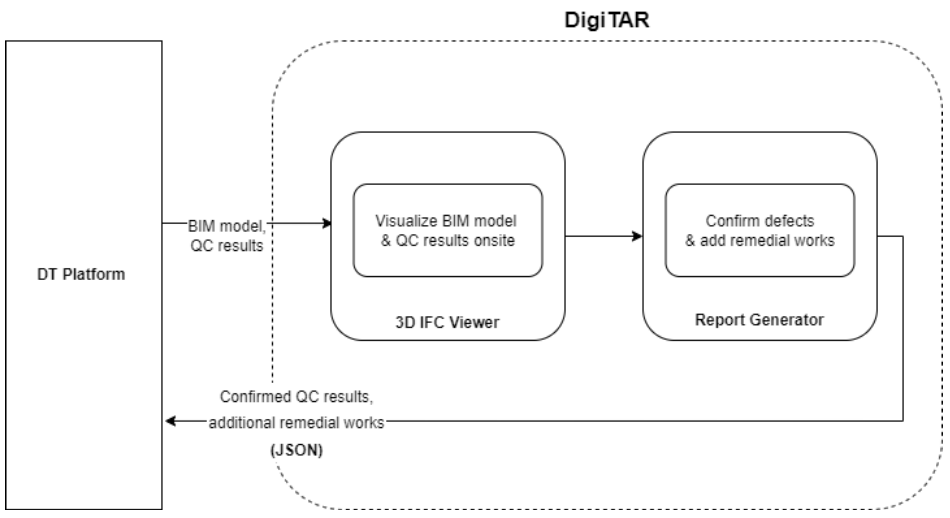

DigiTAR
DT Platform
BIM model, QC results
Visualize BIM model & QC results onsite
3D IFC Viewer
Confirm defects & add remedial works
Report Generator
Confirmed QC results, additional remedial works (JSON)

Figure 1 - DigiTAR Workflow: Quality Control mode

Furthermore, by selecting the second one, the generated H&S results can be also confirmed in-situ and thus the relevant stakeholder can view the potential hazardous regions of the construction site and propose additional mitigations (Figure 2).

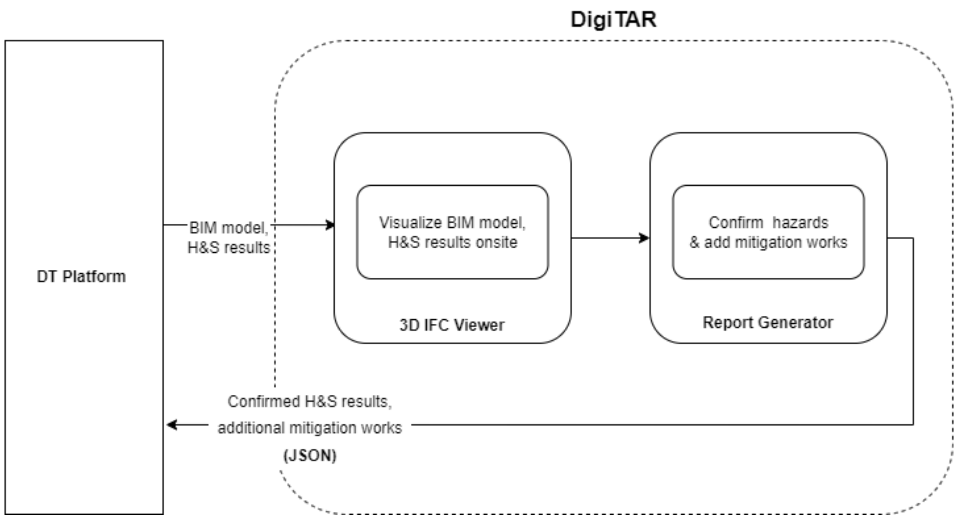


Figure 2 - DigiTAR Workflow: Safety mode

In addition, the Pre-processing mode provides a GUI facilitating data capture (i.e., images) as well as the performance of partial, on-site visual data pre-processing. Based on the work order definition provided by the WODM component, the user can create a new pre-processing job, associate it with specific IFC elements and link the latter with visual data (images) in order to be sent for Visual Quality Control. The DigiTAR module communicates internally with the Visual Data Pre-processing module in order to apply filters to the captured data. The processed data, along with the job metadata, are pushed from the Visual Data Pre-processing module to the DT Platform, where they can be assessed for further QC from the relevant COGITO components (Figure 3).

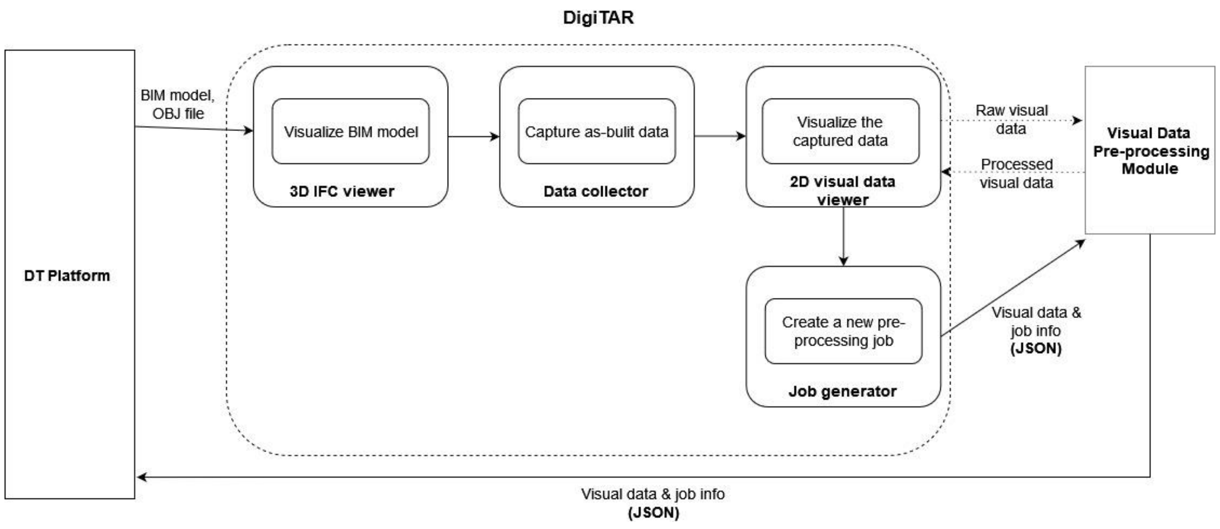


Figure 3 - DigiTAR Workflow: Pre-processing mode

The overall architecture of the DigiTAR tool is depicted in Figure 4. The application consists of three layers: the viewers' layer, the application layer, and the communication layer. The viewers' layer refers to the sub-modules developed for displaying existing data (2D images or 3D BIM models). The application layer includes all the sub-modules that handle the interaction of the user with the application. This layer provides access to the files received by the communication layer (e.g., QC results, H&S results) and displays this information to the users in the form of notifications or in the form of interactive AR menus. This layer is also responsible for handling the input of new data by the user, e.g., the confirmation of the QC results and the suggestion of a remedial work. Finally, the communication layer aims at connecting the DigiTAR module with the DT Platform and the Visual Data Pre-processing module to enable the necessary data exchange across them. All the aforementioned sub-modules and layers are further presented in the next sub-sections.

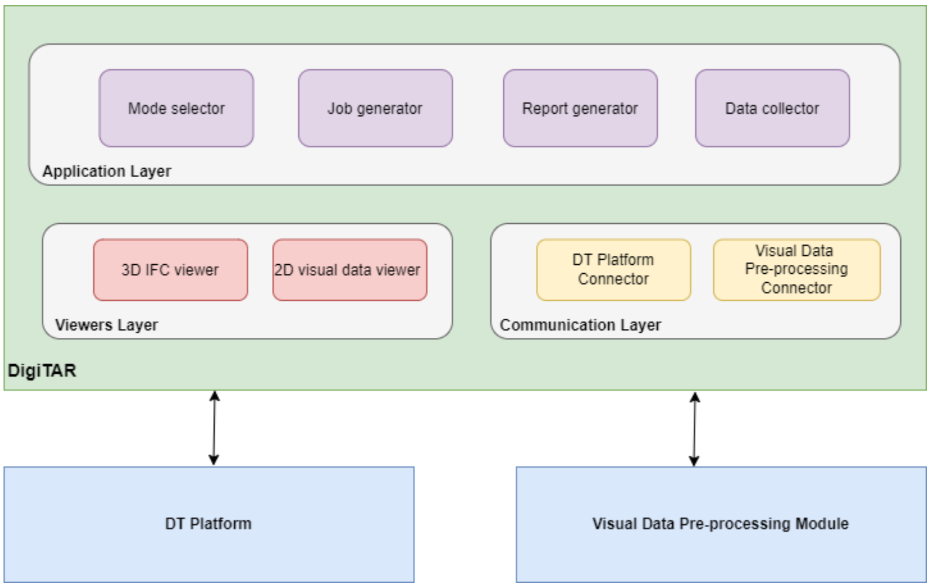


Figure 4 - Overall Architecture of the DigiTAR module

2.1.1 Viewers Layer

2.1.1.1 3D IFC Viewer

The DigiTAR tool is responsible for displaying the 3D BIM model on-site. BIM model visualisation is a key functionality of the application as it is utilised by all three modes of DigiTAR; it is handled by the 3D IFC Viewer sub-module. The architecture of the 3D IFC Viewer component of the DigiTAR tool is graphically illustrated in Figure 5.

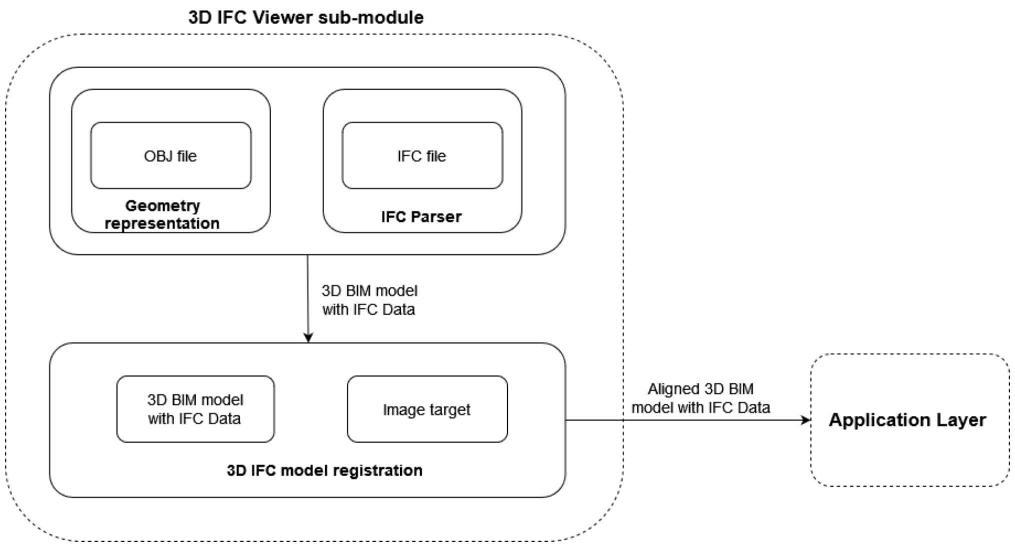

3D IFC Viewer sub-module
OBJ file
Geometry representation
IFC file
IFC Parser
3D BIM model with IFC Data
3D BIM model with IFC Data
Image target
3D IFC model registration
Aligned 3D BIM model with IFC Data
Application Layer

Figure 5 - Architecture of the 3D IFC Viewer component

The development of the DigiTAR tool is based on the Unity 3D Engine, as described in detail in Section 2.2. The Unity Game Engine does not support the generation of the 3D geometry directly from the IFC file. To overcome this obstacle, an OBJ file, that includes the geometric representation of the model, is generated from the IFC, and is imported, along with the IFC, as input to the DigiTAR tool. The 3D IFC Viewer of the DigiTAR tool requires both the geometry representation of the BIM model (OBJ file) and the parsed IFC data:

1. **Geometry Representation**: The Geometry representation of the BIM model is achieved through the transformation of the IFC file to a file format supported by the Unity Game Engine; such a format is the OBJ file format. The DigiTAR receives the OBJ file from the DT Platform, along with the IFC file. Figure 6 presents a visual representation of the OBJ file when imported in Unity.

2. **Parsed IFC Data**: Importing the IFC file into Unity, extracting the IFC data, and mapping those data to the 3D model are handled by custom C# classes based on the Xbim library [1].

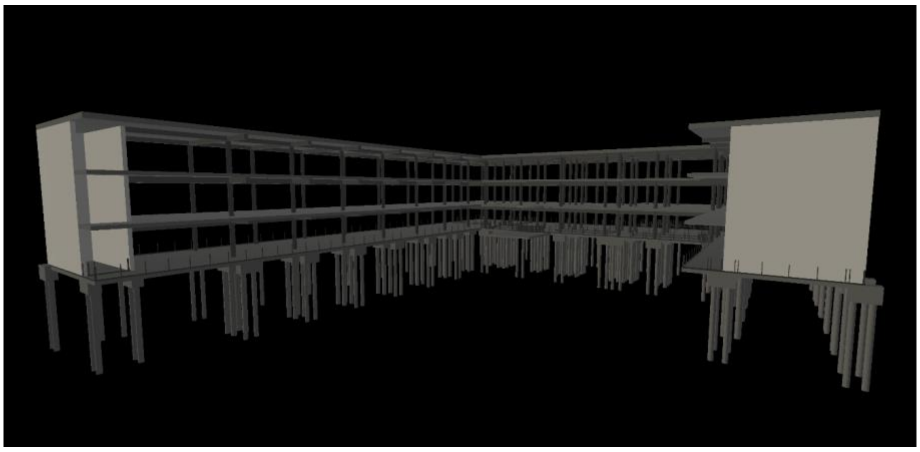


Figure 6 - Visual representation of an OBJ file in Unity

The parsing process of the IFC file takes place in the IFC Parser sub-module. This process is recursive and can be divided in three tasks, as shown in Figure 7. The three subtasks are:

1. **GameObject Creation/Mapping**: In this task, the IFC Parser reads an IFC file, searches for the imported OBJ GameObjects based on the IfcGlobalId, and renames them with the name of the individual IfcElement. Since each IfcElement is uniquely referenced by an IfcGlobalId, each GameObject uniquely references a particular IfcElement. In the case of IFC elements as IfcProject or IfcSite that cannot generate geometry, the IFC Parser creates empty GameObjects with the name of the corresponding IfcElement, so that the hierarchy is complete. In the case of IFC elements as IfcDoor or IfcWall that generate geometry, the GameObjects of these IfcElements exist in the OBJ file (named with the IfcGlobalID of the corresponding IfcElement) and are renamed to hold the name of the corresponding IfcElement.
2. **Metadata Extraction**: In this task, the extraction of properties/materials quantities of each IfcElement takes place. All metadata corresponding to a particular IfcElement as retrieved by the queries are stored in a custom C# class and are added as scripting components to the corresponding GameObject.
3. **Hierarchy Creation**: In this stage, the IFC Parser creates the hierarchy of the BIM model by setting parent-child relationships between the GameObjects. The following relationships are used:

    a. IfcRelAggregates defines the spatial hierarchy between the IFC products,

    b. IfcRelAssignsToGroup associates the IfcSpaces to the IfcZones.

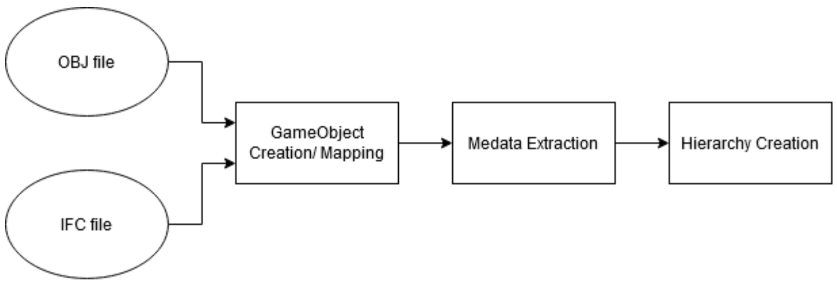


Figure 7 - Process of creating the IFC hierarchy in Unity

The connection between the geometry and the IFC information of the BIM model takes place in the IFC Parser sub-module. The parsing of the IFC and all the necessary data by the IFC Parser, is achieved using LINQ for optimal performance. LINQ [2] stands for a Language-Integrated Query and is a set of technologies based on the integration of query capabilities directly into the C# language. It implements deferred execution, which means it can chain the query statements. Figure 8 shows how the visualisation sub-module interacts with an IFC file through LINQ.

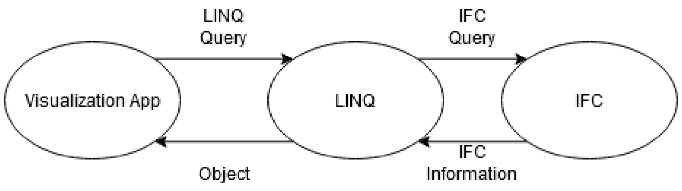
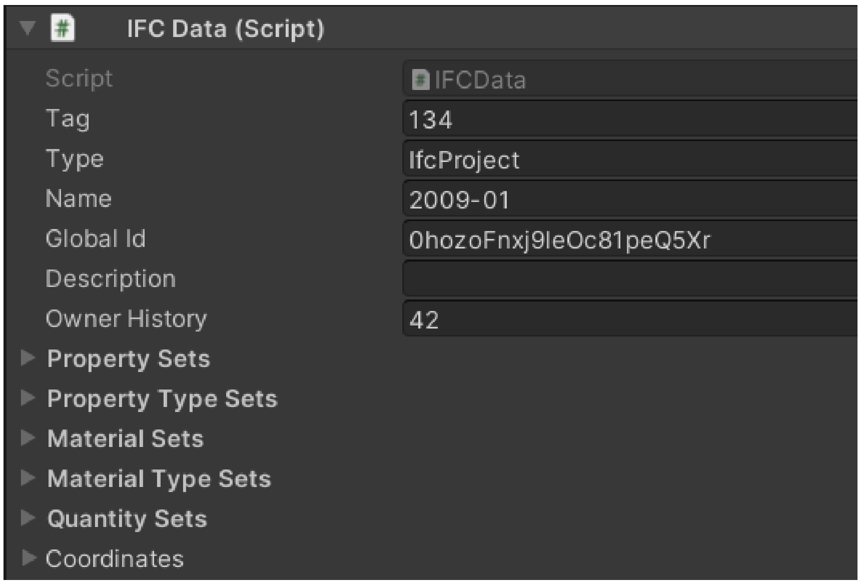


Figure 8 - LINQ to IFC

A query is generated by the IFC Parser and the retrieval of data from the IFC file is accomplished through LINQ. For this process to be successful, the correct query must be made, so knowledge of the IFC Schema is essential.

The purpose of the first query made by the IFC Parser is to find the IfcProject. Then, a recursive process begins:

- An empty GameObject is created and named after the IfcProject; its IFC information is added as an IFC Data Scripting component to the GameObject. In Figure 9, an example of the IFC information included in the form of an IFC Data component is shown.
- Then, through the relationship with other IFC elements the process starts over with the new next IfcElement – as defined in the IFC hierarchy, as a child GameObject to the parent GameObject previously created.
- If a GameObject with the same name as the IfcGlobalId of an IfcElement exists, then it is renamed with the name of the corresponding IfcElement and an IFC Data component with the IFC properties of the IfcElement, is added to the GameObject. The GameObject is set as a child to its parent IfcElement and the procedure is iterated until the whole IFC hierarchy is parsed.



IFC Data (Script)

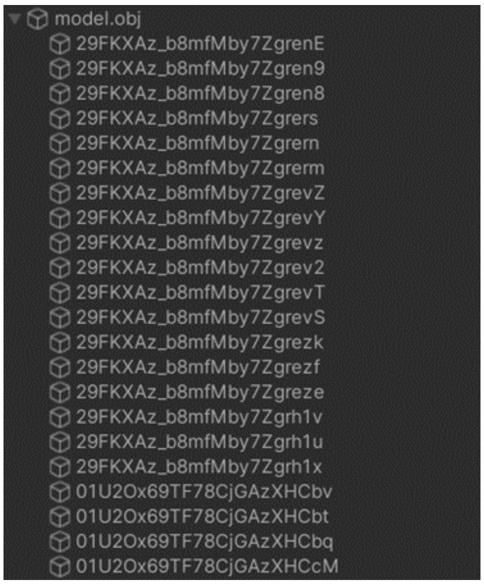
Script	IFCData
Tag	134
Type	IfcProject
Name	2009-01
Global Id	0hozoFnxj9leOc81peQ5Xr
Description	
Owner History	42

▶ Property Sets
▶ Property Type Sets
▶ Material Sets
▶ Material Type Sets
▶ Quantity Sets
▶ Coordinates

Figure 9 - IFC Data component

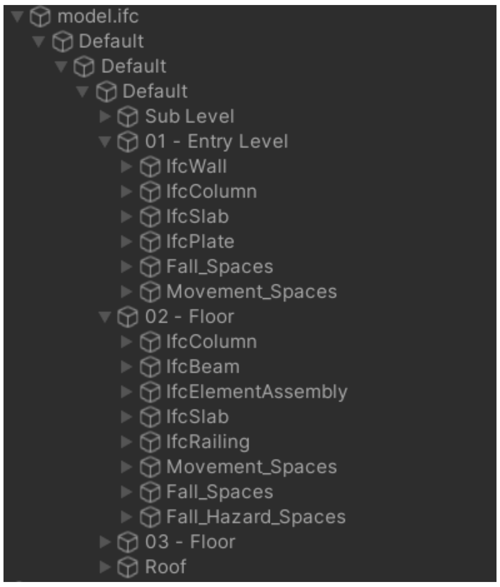
Upon completion of this process, the result is a hierarchical structure, where each GameObject has its own IFC Data component. Also, during this process, for each component a mesh collider is added. Colliders are necessary to interact with the GameObjects, e.g., to select them using HoloLens gestures as discussed in detail in Section 2.4. In case of components that correspond to spaces, e.g., of types IfcSpace and/or

IfcSpatialZone, the colliders that are attached to the corresponding GameObjects are set to convex so that they can collide with other mesh colliders. Collision detection between spaces and the user is useful for user localization in the 3D BIM model, as will be further discussed subsequently in this section. Figure 10 shows the structure of the BIM model in the Unity *Hierarchy* tab as originally imported from the OBJ, while Figure 11 depicts the tree structure which is constructed as a result of the IFC Parser.



```
▼ model.obj
    29FKXAz_b8mfMby7ZgrenE
    29FKXAz_b8mfMby7Zgren9
    29FKXAz_b8mfMby7Zgren8
    29FKXAz_b8mfMby7Zgrers
    29FKXAz_b8mfMby7Zgrern
    29FKXAz_b8mfMby7Zgrerm
    29FKXAz_b8mfMby7ZgrevZ
    29FKXAz_b8mfMby7ZgrevY
    29FKXAz_b8mfMby7Zgrevz
    29FKXAz_b8mfMby7Zgrev2
    29FKXAz_b8mfMby7ZgrevT
    29FKXAz_b8mfMby7ZgrevS
    29FKXAz_b8mfMby7Zgrezk
    29FKXAz_b8mfMby7Zgrezf
    29FKXAz_b8mfMby7Zgreze
    29FKXAz_b8mfMby7Zgrh1v
    29FKXAz_b8mfMby7Zgrh1u
    29FKXAz_b8mfMby7Zgrh1x
    01U2Ox69TF78CjGAzXHCbv
    01U2Ox69TF78CjGAzXHCbt
    01U2Ox69TF78CjGAzXHCbq
    01U2Ox69TF78CjGAzXHCcM
```

Figure 10 - Non-hierarchical structure of imported OBJ in Unity



```
▼ model.ifc
  ▼ Default
    ▼ Default
      ▼ Default
        ▶ Sub Level
        ▼ 01 - Entry Level
          ▶ IfcWall
          ▶ IfcColumn
          ▶ IfcSlab
          ▶ IfcPlate
          ▶ Fall_Spaces
          ▶ Movement_Spaces
        ▼ 02 - Floor
          ▶ IfcColumn
          ▶ IfcBeam
          ▶ IfcElementAssembly
          ▶ IfcSlab
          ▶ IfcRailing
          ▶ Movement_Spaces
          ▶ Fall_Spaces
          ▶ Fall_Hazard_Spaces
        ▶ 03 - Floor
        ▶ Roof
```

Figure 11 - Hierarchical structure created according to the IFC structure

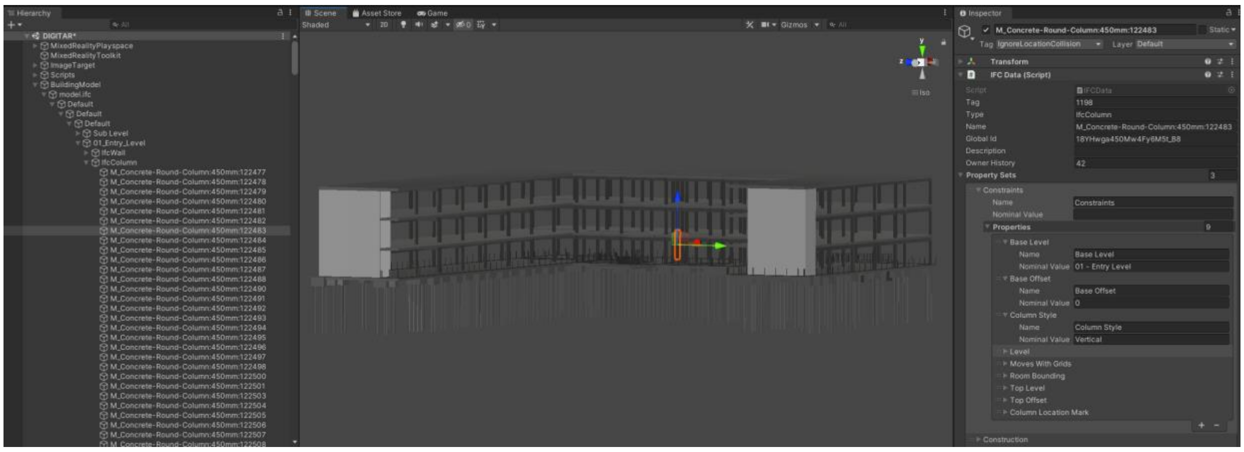
The IFC fields that the IFC Data component stores are listed in Table 1.

Table 1 - IFC properties [3] that the IFC Data component stores.

Property	Description
Tag	The tag identifier at the particular instance of an IFC product.
IfcType	The type of an IfcObject (e.g., IfcSlab, IfcWall).
Name	The name of an IfcObject.
GlobalID	The global unique identifier attribute of an IfcObject.
Description	The description of an IfcObject.
Owner History	The IfcOwnerHistory defines all history- and identification-related information.
Property Sets	The Property Sets for Objects describes how an object occurrence can be related to a single or multiple property sets. A property set contains a single or multiple properties. The data types of an individual property are single value, enumerated value, bounded value, table value, reference value, list value, and combination of property values.
Property Type Sets	The concept template Property Sets for Objects describes how an object type can be related to a single or multiple property sets. A property type set contains a single or multiple properties. The data types of an individual property are the same as the data types of the Property Sets. The property values assigned to an object type apply equally to all occurrences of this object type.
Material Sets	The Material Association concept describes how a product relates to its associated materials that indicates the physical composition of an object. The material association can be of the Association type Material Single, Material Layer Set, Material Layer Set Usage, Material Profile Set, Material Profile Set Usage, and Material Constituent Set.

Material Type Sets	The Material Association concept that describes how a product type relates to its associated materials that indicates the physical composition of an object type. The material type association is the same as the material association in the Material Sets.
Quantity Sets	Contain all multiple quantity occurrences that relate to an object. The data type of quantity occurrence values is count, length, area, volume, weight, time, or a combination of quantities.
Coordinates	Refers to the world coordinates of an IfcObject calculated from the relative coordinates of the Project.

Figure 12 shows an example of a BIM model, as seen in the Unity Editor Scene. In this example, a particular component (a column) is selected. The created IFC hierarchy is depicted on the left (in the *Hierarchy* of the scene), while the IFC properties for the selected object are displayed on the right (in the *Inspector* section). As can be seen, the IFC Data component stores and displays the IFC properties for the selected object as listed in Table 1. It should be noted that as the construction process evolves, the BIM models visualised within the DigiTAR tool will be updated to mirror the current as-build status of the construction sites.

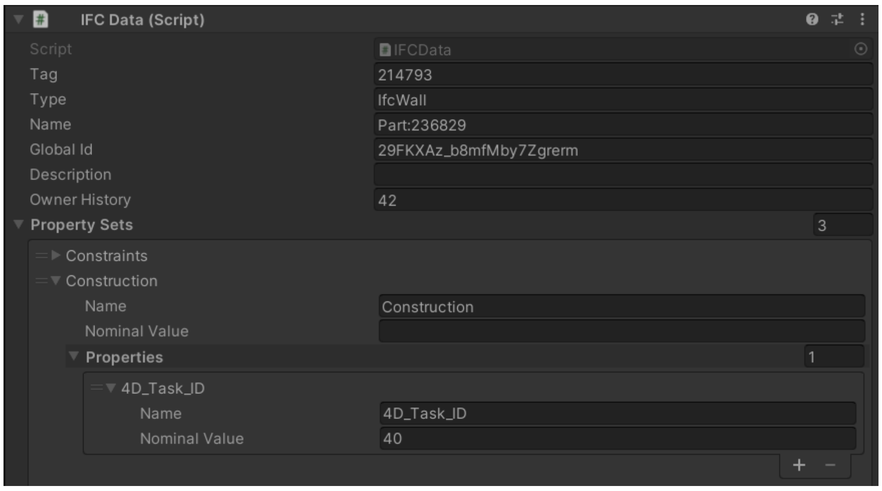


```
DIGITAR*
  MixedRealityPlayspace
  MixedRealityToolkit
  ImageTarget
  Scripts
  BuildingModel
    model.ifc
      Default
        Default
          Default
            Sub Level
            01_Entry_Level
              IfcWall
              IfcColumn
                M_Concrete-Round-Column:450mm:122477
                M_Concrete-Round-Column:450mm:122478
                M_Concrete-Round-Column:450mm:122479
                M_Concrete-Round-Column:450mm:122480
                M_Concrete-Round-Column:450mm:122481
                M_Concrete-Round-Column:450mm:122482
                M_Concrete-Round-Column:450mm:122483
                M_Concrete-Round-Column:450mm:122484
                M_Concrete-Round-Column:450mm:122485
                M_Concrete-Round-Column:450mm:122486
                M_Concrete-Round-Column:450mm:122487
                M_Concrete-Round-Column:450mm:122488
                M_Concrete-Round-Column:450mm:122490
                M_Concrete-Round-Column:450mm:122491
                M_Concrete-Round-Column:450mm:122492
                M_Concrete-Round-Column:450mm:122493
                M_Concrete-Round-Column:450mm:122494
                M_Concrete-Round-Column:450mm:122495
                M_Concrete-Round-Column:450mm:122496
                M_Concrete-Round-Column:450mm:122497
                M_Concrete-Round-Column:450mm:122498
                M_Concrete-Round-Column:450mm:122500
                M_Concrete-Round-Column:450mm:122501
                M_Concrete-Round-Column:450mm:122503
                M_Concrete-Round-Column:450mm:122504
                M_Concrete-Round-Column:450mm:122505
                M_Concrete-Round-Column:450mm:122506
                M_Concrete-Round-Column:450mm:122507
                M_Concrete-Round-Column:450mm:122508
```

```
M_Concrete-Round-Column:450mm:122483              Static
Tag IgnoreLocationCollision     Layer Default
Transform
IFC Data (Script)
  Script            IFCData
  Tag               1198
  Type              IfcColumn
  Name              M_Concrete-Round-Column:450mm:122483
  Global Id         18YHwga450Mw4Fy6M5t_B8
  Description
  Owner History     42
Property Sets                                   3
  Constraints
    Name            Constraints
    Nominal Value
    Properties                                  9
      Base Level
        Name            Base Level
        Nominal Value   01 - Entry Level
      Base Offset
        Name            Base Offset
        Nominal Value   0
      Column Style
        Name            Column Style
        Nominal Value   Vertical
      Level
      Moves With Grids
      Room Bounding
      Top Level
      Top Offset
      Column Location Mark
  Construction
```

Figure 12 - Column selection

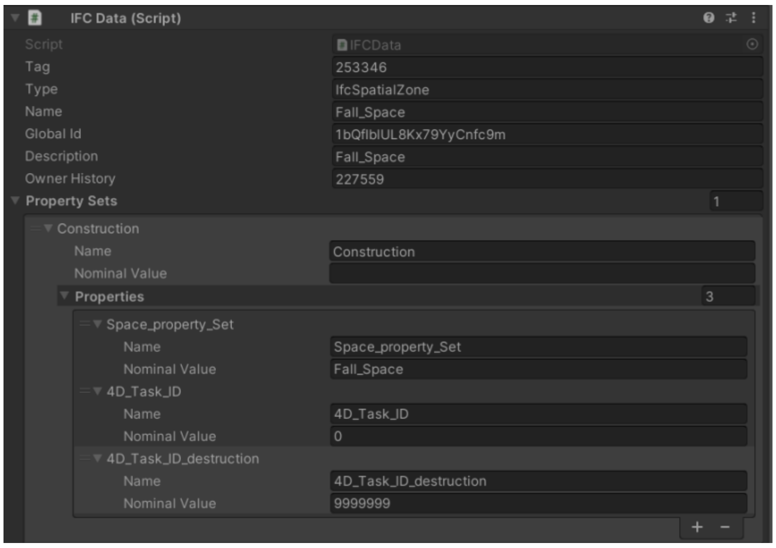
As mentioned in the architecture in Section 2.1 and graphically illustrated in Figure 2, for the Safety mode, the DigiTAR tool receives and parses an IFC that is enriched with safety information. The IFC Parser is extended to parse these safety-related properties for each element. The IFC Data component with the injected safety properties for an existing IFC element and for a new SafeCon (SC) object (that corresponds to an IfcSpatialZone) are depicted in Figure 13 and Figure 14, respectively. As can be seen, the safety related property set is called "Construction" and each element has a property named "4D_Task_ID" and a property named "4D_Task_ID_destruction". The first attribute corresponds to the element that created the new SC object, and the second attribute corresponds to the 4D_task_ID of the element that removed the SC object. The second property represents the temporary nature of the elements that are introduced and are later removed in the BIM model. Detailed description of the enriched with safety information IFC file is included in D4.2 "Preventive Health & Safety Application v2".



IFC Data (Script)

Script	IFCData
Tag	214793
Type	IfcWall
Name	Part:236829
Global Id	29FKXAz_b8mfMby7Zgrerm
Description	
Owner History	42
Property Sets	3
Constraints	
Construction	
Name	Construction
Nominal Value	
Properties	1
4D_Task_ID	
Name	4D_Task_ID
Nominal Value	40

Figure 13 - The IFC Data component with the injected safety properties for an existing IFC element



IFC Data (Script)

Script	IFCData
Tag	253346
Type	IfcSpatialZone
Name	Fall_Space
Global Id	1bQfIbIUL8Kx79YyCnfc9m
Description	Fall_Space
Owner History	227559
Property Sets	1
Construction	
Name	Construction
Nominal Value	
Properties	3
Space_property_Set	
Name	Space_property_Set
Nominal Value	Fall_Space
4D_Task_ID	
Name	4D_Task_ID
Nominal Value	0
4D_Task_ID_destruction	
Name	4D_Task_ID_destruction
Nominal Value	9999999

Figure 14 - The IFC Data component with the injected safety properties for a new SafeCon (SC) object (that corresponds to an IfcSpatialZone)

In this version of the DigiTAR tool, refinements have been applied to the visual appearance of the 3D BIM model compared to the first version of the tool in order to enhance the user experience. As mentioned before, the DigiTAR tool receives as an input the geometry of the model in an OBJ file. In the OBJ file, the geometry of each IFC element is represented by a mesh. In Unity, **rendering** is the process of drawing all the objects that are included in the Unity scene on the AR device's screen. Each mesh included in the OBJ file is rendered by a *Mesh Renderer Component*. The way that a mesh is rendered depends on its *Materials, Shaders,* and *Textures* [4]. In the first version of the DigiTAR tool, the Unity Standard Shader was utilized in the 3D BIM model visualisation. In this version of the tool, the Mixed Reality Toolkit (MRTK) Standard Shader (see Section 2.2 for details on MRTK) is utilized for rendering the 3D BIM model. In Figure 15 and Figure 16, the same scene view is depicted as rendered in Unity Editor and in the AR HMD device (HoloLens) using the Unity Standard Shader and the MRTK Standard Shader, respectively. As can be seen, the MRTK Shader provides more realistic, mixed reality view of the model while also having the advantage of being computationally efficient for mixed reality devices. We also experimented with transparent shaders but concluded that the transparent representation overburdened the users' on-site orientation since the 3B BIM model visualisation was not a physical representation of their environment. In Figure 17, the scene rendering in Unity Editor and in the AR HMD device (HoloLens) using a transparent shader is depicted.

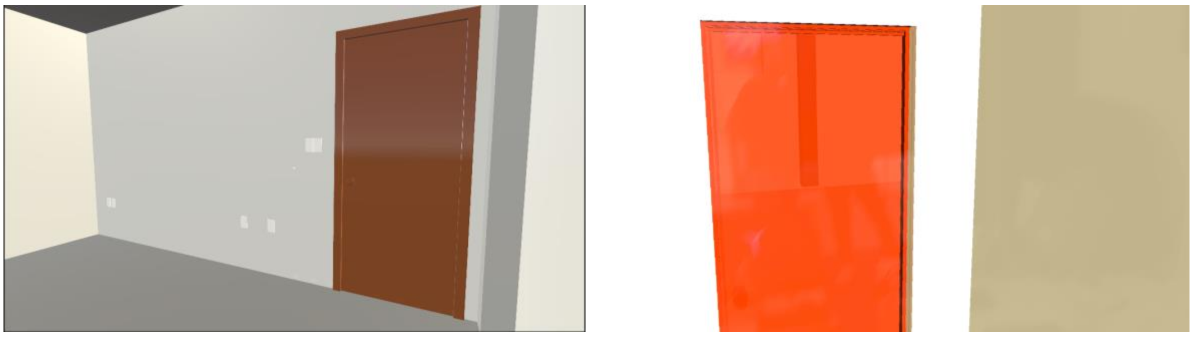


Figure 15 - Scene rendering in Unity Editor (left figure) and in the AR HMD (HoloLens) (right figure) using the Unity Standard Shader

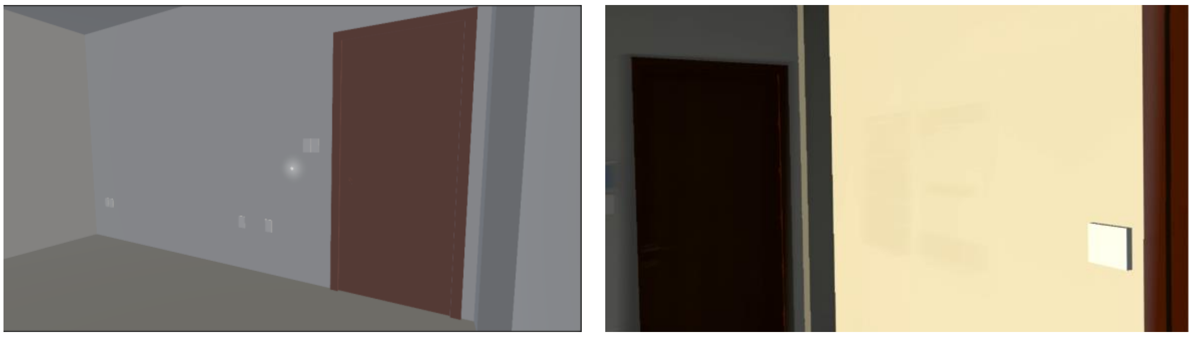


Figure 16 - Scene rendering in Unity Editor (left figure) and in the AR HMD (HoloLens) (right figure) using the MRTK Standard Shader

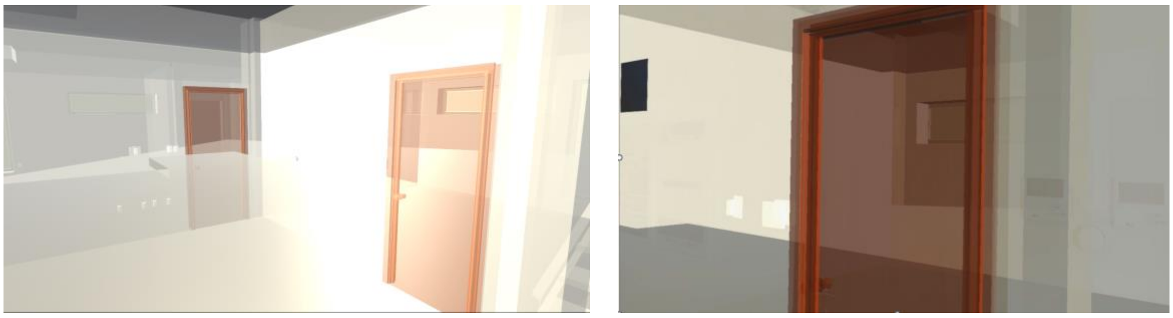


Figure 17 - Scene rendering in Unity Editor (left figure) and in the AR HMD (HoloLens) (right figure) using the transparent Unity Shader (here, the objects are 35% transparent)

It should be noted that the loading and parsing of the 3D BIM model are done automatically within the DigiTAR tool based on the project that the user selects in the corresponding menu (see Figure 60). After loading the 3D BIM model, the next step is to align the 3D model of the site to the actual site. This procedure is called **registration**. The registration progress is performed manually by the user when the 3D BIM model is firstly visualised. Within the DigiTAR application, registration relies on image targets. An image target is an image that the application running on the HoloLens will detect and track. This image will be the link between the static 3D world (BIM model) and the real world.

The image target is printed and positioned at a location in the real world, ensuring that it is accessible to the person wearing the HoloLens. At the same time, an identical image is placed in exactly the same spot in the 3D BIM model. To enable the detection of the image target, the user should use speech command "Scan". This way, the data that are captured by the HoloLens sensors and cameras are utilised for image target detection. More specifically, features are extracted from the HoloLens camera stream and are compared to the reference features already extracted from the image target. In the context of pattern recognition, the features that are extracted in advance from the image target constitute the pattern that the algorithm searches across the continuous flow of data streams. When the person wearing the HoloLens looks at the image target, the features extracted from the data stream of HoloLens are matched to the pattern of features belonging to the image target. Therefore, the image target is detected. The image target used in the DigiTAR application, as well as an illustration of the extracted features, are depicted in Figure 18.

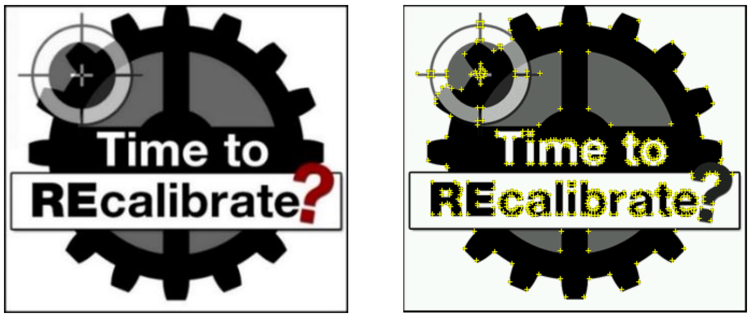


Figure 18 - The image target used in the DigiTAR application, along with an illustration of the extracted features that are detected by the HoloLens camera

After successful registration of the 3D BIM model and in order to maintain it, the registered 3D BIM model is continuously tracked. In the DigiTAR application, the registration of the 3D BIM model is tracked using **spatial anchors**; spatial anchors represent important points in the world that the HoloLens coordinate system keeps track of over time. The registered 3D BIM model can be set as a spatial anchor using the speech command “Anchor”. This way, the next time a user opens the DigiTAR application, the 3D BIM model is loaded aligned to the real world without the need to repeat the registration process. The workflow of the registration process is graphically illustrated in Figure 19. When the user uses speech command “Scan” again, the spatial anchor is deleted so that the model’s position can be redetermined upon image target detection. To anchor the new alignment, the user can use speech command “Anchor”. The complete pipeline of how the user of the DigiTAR tool performs the registration process is described in detail in Section 2.4.2, that encloses a complete guide to the various functionalities of the DigiTAR tool.

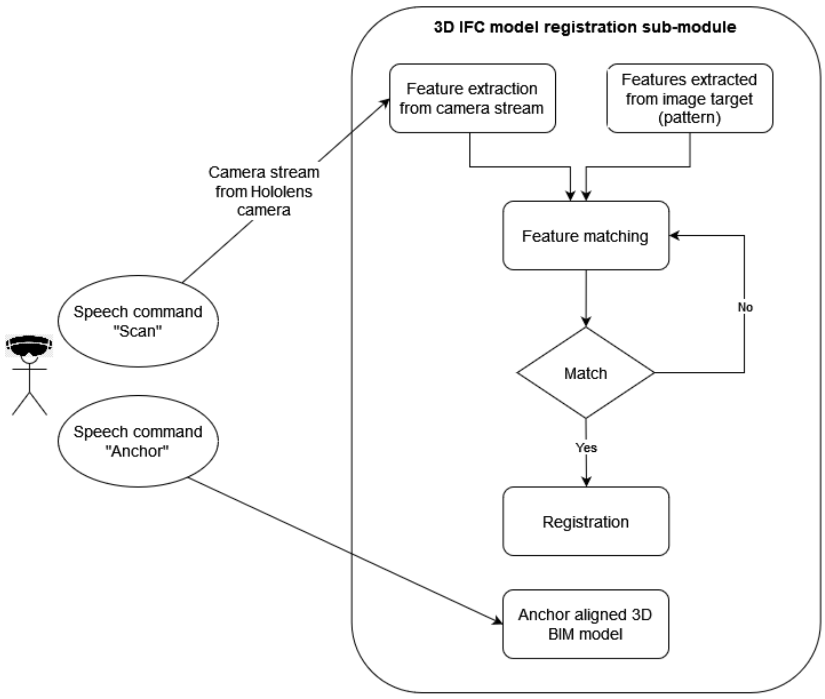
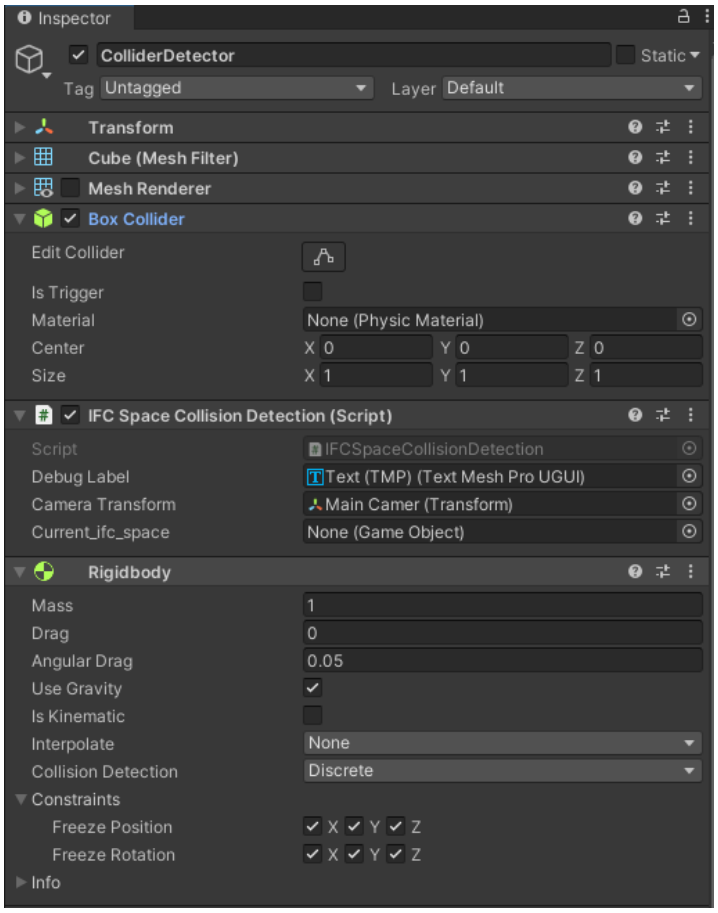

3D IFC model registration sub-module
Feature extraction from camera stream
Features extracted from image target (pattern)
Feature matching
Match
No
Yes
Registration
Anchor aligned 3D BIM model
Camera stream from Hololens camera
Speech command "Scan"
Speech command "Anchor"

Figure 19 – 3D BIM registration process

Registration is also important for successful **localization** of the user in the 3D BIM model. Knowing the position and/or orientation of the user in the 3D BIM model is necessary for the pre-processing jobs created within the DigiTAR tool. As explained in detail in Section 2.1.2.2, the capture device specification is a necessary step to create a pre-processing job. Within the DigiTAR, the capture device is always the HoloLens device, which the user wears on their head and on which the application runs. The location and orientation of the capture device, i.e., the HoloLens, are essential properties included in the device specification for each pre-processing job. To locate the user in the 3D BIM model, the relative position of the user to the origin of the IFC model is calculated within the DigiTAR tool. This localization is combined with the georeferenced coordinates that will also be available through the DT Platform. The position and orientation of the user are computed and included in the JSON file that is sent to the Visual Data Pre-processing module, during the creation of a pre-processing job within DigiTAR.

In DigiTAR, it is useful to determine when the user enters and/or exits an IfcSpace and/or an fcSpatialZone. This is especially important for the Safety mode where the user should be notified when entering hazardous areas. In DigiTAR, we rely on collision detection to determine when the user enters and/or exits an IfcSpace or an IfcSpatialZone. As mentioned before, convex mesh colliders are attached to the GameObjects that correspond to spaces to enable their collision with other colliders. In applications created for HMD, such as the DigiTAR tool, the position of the MainCamera corresponds to the user's position in the model. By creating a new GameObject (namely "ColliderDetector" GameObject), setting it as a child to the MainCamera, and attaching a Rigidbody component to it, its motion is controlled by the Unity's physics engine and the collisions between its collider and other convex mesh colliders are detected. Collision detection is properly handled in a script (i.e., "IfcSpaceCollistionDetection" script) attached to the "ColliderDetector". This way, we can detect when the user enters and/or exits a space, i.e., determine user's position in terms of IfcSpace and/or IfcSpatialZone. In Figure 20, the scripts attached to the "ColliderDetector" GameObject are depicted. As can be seen, the GameObject has the "Rigidbody" script and the "IfcSpaceCollisiionDetection" script attached. The first enables collision detection while the latter handles how the collisions between the "ColliderDetector" and the convex colliders that belong to IfcSpaces and/or IfcSpatialZones are handled within DigiTAR.


Inspector

✓ ColliderDetector Static
Tag Untagged Layer Default

Transform
Cube (Mesh Filter)
Mesh Renderer
✓ Box Collider

Edit Collider	
Is Trigger	
Material	None (Physic Material)
Center	X 0 Y 0 Z 0
Size	X 1 Y 1 Z 1

✓ IFC Space Collision Detection (Script)

Script	IFCSpaceCollisionDetection
Debug Label	Text (TMP) (Text Mesh Pro UGUI)
Camera Transform	Main Camer (Transform)
Current_ifc_space	None (Game Object)

Rigidbody

Mass	1
Drag	0
Angular Drag	0.05
Use Gravity	✓
Is Kinematic	
Interpolate	None
Collision Detection	Discrete
Constraints	
Freeze Position	✓ X ✓ Y ✓ Z
Freeze Rotation	✓ X ✓ Y ✓ Z
Info	

Figure 20 – The "ColliderDetector" GameObject as seen in the *Inspector* in Unity

2.1.1.2 2D Visual Data Viewer

Within the 2D Visual Data Viewer, the user can view visual data, both raw and processed. The user can either capture photos at runtime or select photos from the HoloLens gallery and attach them to a pre-processing job. In both cases, the user can view the attached photos within the 2D Visual Data Viewer component of the application. The raw images (either captured at runtime or selected from the gallery) are displayed in the *Preview* menu of the application. The *Preview* menu also provides the option to select a filter and apply it to the raw image. Finally, the pre-processing job, the filter specification and the raw photo are sent to the Visual Data Pre-processing module, which returns the processed image back to the DigiTAR when the filtering is applied. The processed image can be viewed in the *Preview* menu. This way, the user can select different filters and preview the processed images in the *Preview* menu. When the user is satisfied with the processed image, they can select to finalise the pre-processing job, i.e., send it to the DT Platform. Figure 21 illustrates the basic workflow for capturing and/or selecting raw photos, selecting filters and finally, receiving the processed data (from the Visual Data Pre-processing tool) in the DigiTAR tool.

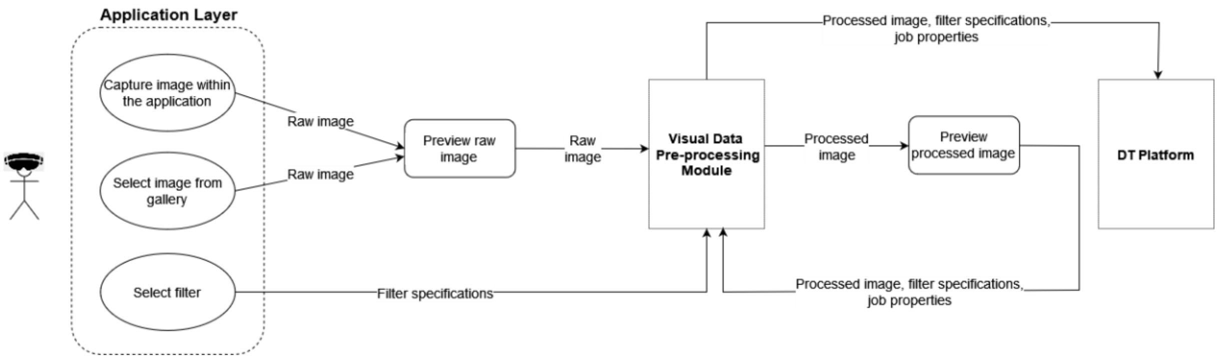

Application Layer
Capture image within the application
Select image from gallery
Select filter
Raw image
Raw image
Preview raw image
Raw image
Visual Data Pre-processing Module
Processed image
Preview processed image
DT Platform
Processed image, filter specifications, job properties
Filter specifications
Processed image, filter specifications, job properties

Figure 21 - Workflow for the 2D Visual Data Viewer of the DigiTAR tool

2.1.2 Application Layer

2.1.2.1 Mode Selector

Within the mode selector, the user can choose between the three different modes of DigiTAR: (i) the visualisation of the Quality Control results (Geometric and Visual), (ii) the visualisation of the H&S results and finally (iii) the Pre-processing mode. Figure 22 illustrates the mode selection process.

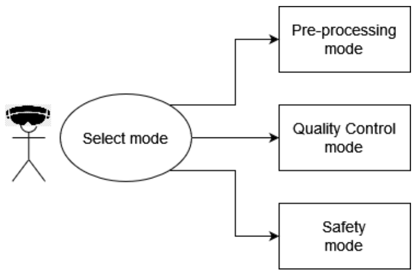


Figure 22 - Selecting mode in the DigiTAR tool

2.1.2.2 Job Generator

Using this sub-module, the user can generate new jobs and link them with new data; the jobs are ultimately submitted for pre-processing to the Visual Data Pre-processing module. The Visual Data Pre-processing module is responsible for uploading the pre-processing jobs created within the DigiTAR to the DT platform so that they can be used for defect detection.

The DigiTAR user navigates on-site through the 3D BIM model (which is aligned to the actual site due to the registration process, described in Section 2.1.1.1). Figure 23 presents the generation of a pre-processing job within the DigiTAR tool. The user of the DigiTAR tool can create a new pre-processing job and associate it with an IFC element. When creating a pre-processing job, the capture device properties should be specified.

The 3D IFC Viewer sub-module of the DigiTAR tool determines the current position and orientation of the capture device (i.e., HoloLens) in the 3D space. The location and orientation information, along with other device properties, such as the device ID, are included in the metadata file (in JSON format) that is pushed to the Visual Data Pre-processing module. Properties that are also included in the generated JSON file are the name of the job, the image format (e.g., ".png"), the image filename, and the time that the user created the job.

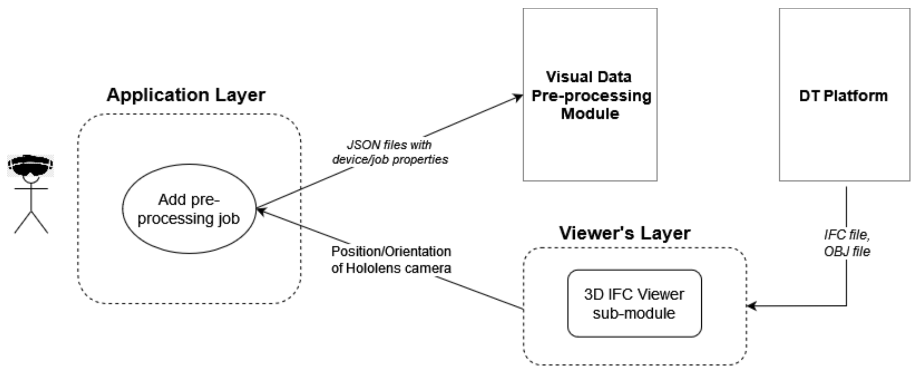

Application Layer
Add pre-processing job
JSON files with device/job properties
Visual Data Pre-processing Module
DT Platform
Position/Orientation of Hololens camera
Viewer's Layer
3D IFC Viewer sub-module
IFC file, OBJ file

Figure 23 – Pre-processing job generation within the DigiTAR tool

2.1.2.3 Report Generator

This sub-module generates confirmation reports or additional remedial works and mitigations that are going to be delivered to the DT Platform. For the Quality Control mode, the user firstly views the Quality Control results and selects whether to confirm or reject them. If the user selects to confirm the QC result, the DigiTAR tool prompts the user to provide a remedial work. The same procedure applies for the Safety Control mode, where the user can confirm the H&S results and provide mitigation works.

2.1.2.4 Data Collector

The Data Collector is in charge of capturing on-site raw data that is going to be sent to the Visual Data Pre-processing module for pre-processing with filter application. There are two workflows for capturing raw data on-site within the Pre-processing mode of the DigiTAR tool; 1) the user can create a pre-processing job and select to capture an image to be linked to the job, 2) the user can capture images without relating them to a job. In the second workflow, the images are saved to the HoloLens gallery with metadata that include the position and orientation of the user in the 3D BIM model at capture time. Thus, the user can select these images at subsequent sessions and link them to a job. Figure 24 presents the workflow for the data collection functionality of the DigiTAR tool.

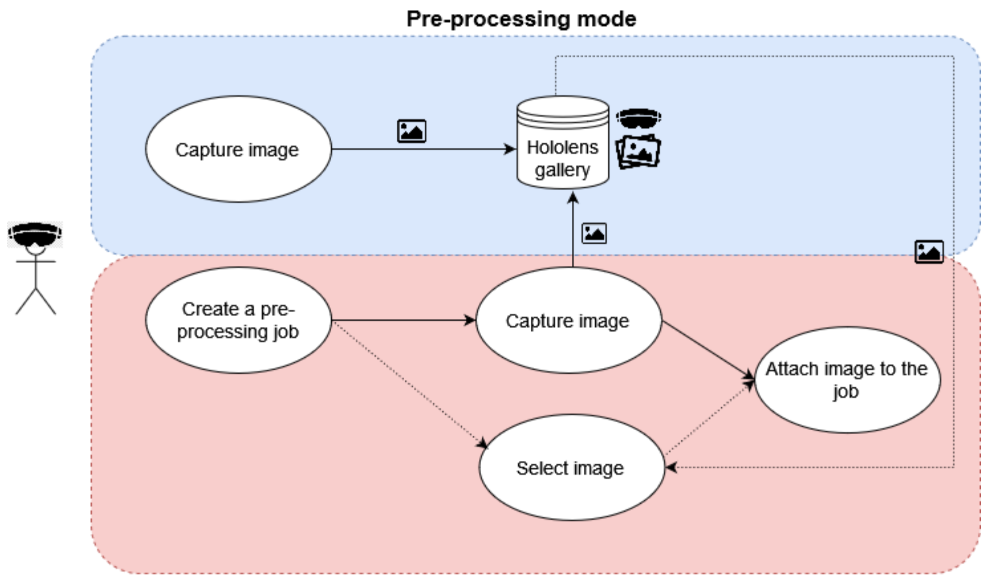


Figure 24 - Workflow for the Data Collector component of the DigiTAR tool

2.1.3 Communication Layer

2.1.3.1 DT Platform Connector

The DT Platform Connector enables the communication with the DT Platform. This component is responsible for uploading and downloading files and relative metadata through the DT Platform API. Communication is achieved using the DT Platform's API, as described in Section 2.3.3.

2.1.3.2 Visual Data Pre-processing Connector

The Visual Data Pre-processing Connector handles the direct communication between the DigiTAR tool and the Visual Data Pre-processing tool. Communication is achieved using the Visual Data Pre-processing API, which is described in detail in Section 2.3.3.

2.2 Technology Stack and Implementation Tools

The DigiTAR application is developed from scratch. The development is performed using the Unity3D engine and the C# programming language. Several libraries and packages, such as the Mixed Reality Toolkit (MRTK), are used for the development of the application (see Table 2). MRTK enables access to the HoloLens' capabilities and sensors. During the implementation phase, testing was merely performed on a HoloLens2 device by building and deploying the application within Unity and Visual Studio. Alternatively, testing is always possible on the official HoloLens emulator provided by Microsoft.

Unity3D: Unity [5] is a Game Engine created by Unity Technologies. The engine is widely used to build 3D and 2D applications, simulations, and games as well as Virtual Reality (VR) and AR applications.

Programming Language C#: Programs based on/around Unity can be written in the object-oriented programming language C#. C# enables the development of a variety of secure and robust applications that run in the .NET ecosystem. The scripts to implement the DigiTAR application are developed in C#.

Microsoft HoloLens2: The necessary hardware for the DigiTAR application is a pair of smart glasses with all the necessary sensors and AR capabilities. Microsoft HoloLens2, depicted in Figure 25, is a pair of mixed reality smart glasses developed and manufactured by Microsoft [6]. Microsoft HoloLens is the first fully self-contained holographic computer; the users can move freely, with no wires or external packs needed.

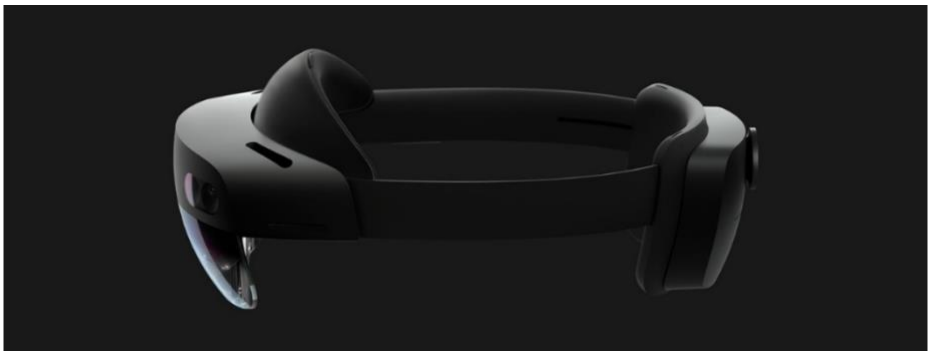


Figure 25 - Microsoft HoloLens smart glasses [6]

Mixed Reality Toolkit: Microsoft provides substantial support for both users and developers regarding its HoloLens product; the Mixed Reality Toolkit (MRTK) [7] for Unity provides essential sets of components and features for developing applications for Microsoft HoloLens 1st gen and HoloLens2. MRTK provides useful interfaces and implementations for creating Mixed Reality applications, accompanied by full documentation and support.

HoloLens Emulator: To facilitate the development of HoloLens applications, Windows Mixed Reality Software Development Kit (SDK) provides the HoloLens Emulator [8]. The HoloLens Emulator enables the testing of holographic applications on a computer without the need for a physical HoloLens product. The human and environmental inputs that are usually read by HoloLens sensors are simulated from keyboard, mouse, or Windows Mixed Reality motion controllers [9]. Notably, applications running on the emulator do not need to be built from scratch to be deployed on the HoloLens device. The main window of HoloLens Emulator is depicted in Figure 26.

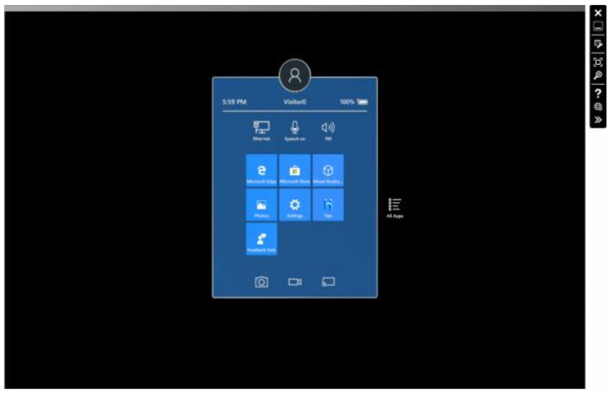


Figure 26 - Main window of HoloLens2 Emulator [8]

Table 2 – Libraries and Technologies used in User interface and AR enabled in-situ QC Visualisation v2

Library/Technology Name	Version	License
Xbim toolkit	5.1.323	Common Development and Distribution License (CDDL)
OBJ Runtime Importer	2.02	Extension Asset
Microsoft HoloLens	2nd gen	-
Windows Mixed Reality	SDK 10.0.19041	Software licensing
Unity 3D Editor	2020.5	Software licensing
Mixed Reality Toolkit (MRTK)	2.5.3	Software licensing

Vuforia Engine	9.6.4	Proprietary

2.3 Input, Output and API Documentation

The DigiTAR module interacts with the DT Platform and the Visual Data Pre-processing module for data exchange. The input data and output data for each mode of the DigiTAR tool are presented in Sections 2.3.1 and 2.3.2, respectively. Finally, the API is documented in Section 2.3.3.

2.3.1 Input Data

DigiTAR, when used in Quality Control mode, would require the following data as input:

- **Project ID:** the project ID that the user wants to check.
- **QC results:** a file in JSON format containing all the information generated by the relevant COGITO QC components
- **IFC file:** the BIM model exported in IFC and in OBJ file formats in order to connect the as-designed data and the QC results generated by the respective COGITO QC components.

In the case of Safety mode, DigiTAR would require the following data as input:

- **Project ID:** the project ID that the user wants to check.
- **IFC file enriched with Safety information**: BIM model exported in IFC and in OBJ file formats, enriched with the information generated by the COGITO Health and Safety-related components.

Finally, in the case of operation in Pre-processing mode, DigiTAR would require the following data as input:

- **Project ID:** the ID of the project the user wants to check.
- **IFC file:** BIM model exported in IFC and in OBJ file formats in order to connect the as-designed data and the as-build data (IFC components' attribution).
- **Work Order ID:** the working order ID to associate the image data and extract the list of components that are ready for quality control.

2.3.2 Output Data

The DigiTAR module in the case of Quality Control mode would provide the following data as output:

- **Metadata:** a file in JSON format containing information regarding the defects' confirmation and the additional remedial works.

In the case of Safety mode, DigiTAR would provide the following data as output:

- **Metadata:** a file in JSON format containing information regarding the hazards' confirmation and the additional mitigation works.

In the case of Pre-processing mode, DigiTAR would provide the following data as output:

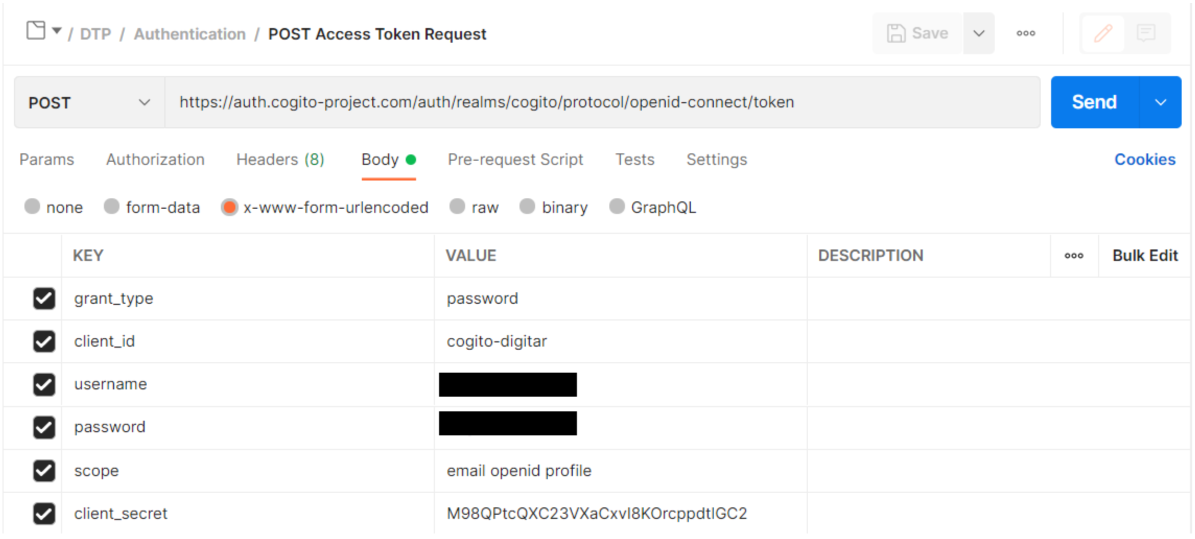
- **Raw and processed as-built 2D image data:** processed images (i.e., .png, .jpeg) of the as-built data that will be utilised by the Visual Quality Control tool.
- **Metadata:** files in JSON format containing information accompanying the processed image, the capturing device properties, and the pre-processing job properties.

2.3.3 API Documentation

Regarding the API, a second version is implemented in this final release of the User interface and AR enabled In-situ QC Visualisation. The API is mainly used for storing, retrieving, updating, and deleting data used within the DigiTAR application. In this second version of the DigiTAR tool, the execution interaction of the DigiTAR tool with the DT Platform is established. Following the DT Platform API documentation and to be consistent with the rest of the COGITO solution, specific endpoints are created for each interaction with the DT Platform. Firstly, communication for user authentication through Keycloak is established. Also, requests are established to the DT platform for retrieving the list of available projects and for acquiring the relevant files for each project, i.e., the IFC file and the OBJ file. The API is also used for communicating with the Visual Data Pre-processing module. To this end, the API is used to send raw images to the Visual Data Pre-processing module and receive the pre-processed images along with metadata. A few examples of all the available requests are presented in the next sub-sections. The requests were firstly tested using the Postman API Platform[1] and were subsequently integrated within the DigiTAR tool.

2.3.3.1 User Authentication Request

When the user of the DigiTAR tool inserts their credentials in the application, a request is made to the Keycloak Account Management System for user authentication. In this request, the DigiTAR tool (like each COGITO tool) has its own "client_id" and "cliend_secret" values. The API endpoint for validating the user's credentials and retrieving the user's access token is depicted in Figure 27. After successfully retrieving the access token for the specific user, this token is used as a query parameter in another request to retrieve information regarding the user, as depicted in Figure 28. The user ID retrieved by the latter query can be used to query the user's roles, as depicted in Figure 29.

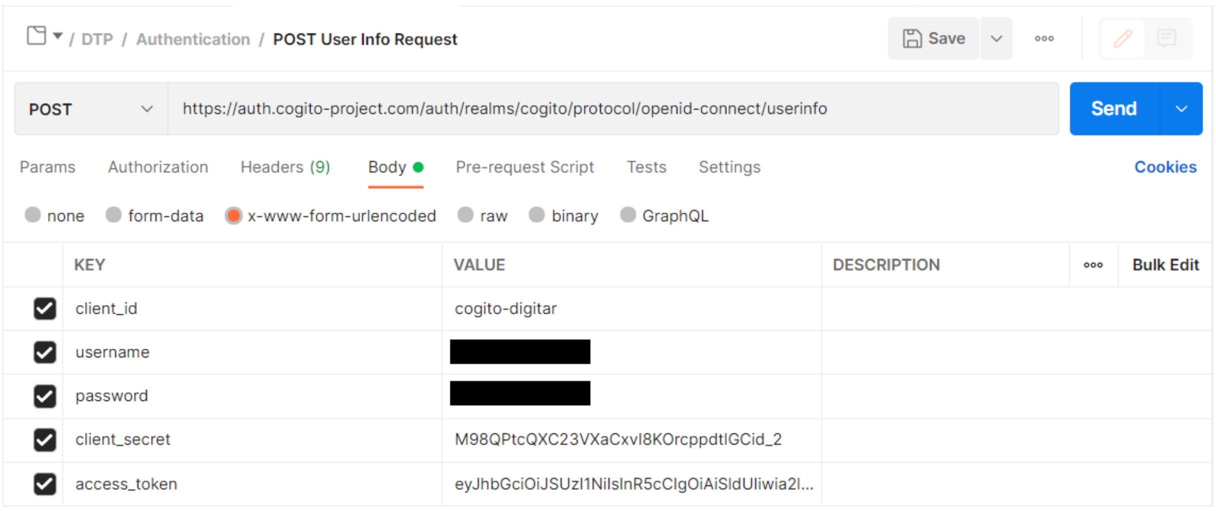


POST https://auth.cogito-project.com/auth/realms/cogito/protocol/openid-connect/token

	KEY	VALUE	DESCRIPTION
✓	grant_type	password	
✓	client_id	cogito-digitar	
✓	username	[redacted]	
✓	password	[redacted]	
✓	scope	email openid profile	
✓	client_secret	M98QPtcQXC23VXaCxvI8KOrcppdtIGC2	

Figure 27 - User authentication request

[1] https://www.postman.com/[2] https://docs.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/using-the-windows-device-portal#setting-up-hololens-to-use-windows-device-portal

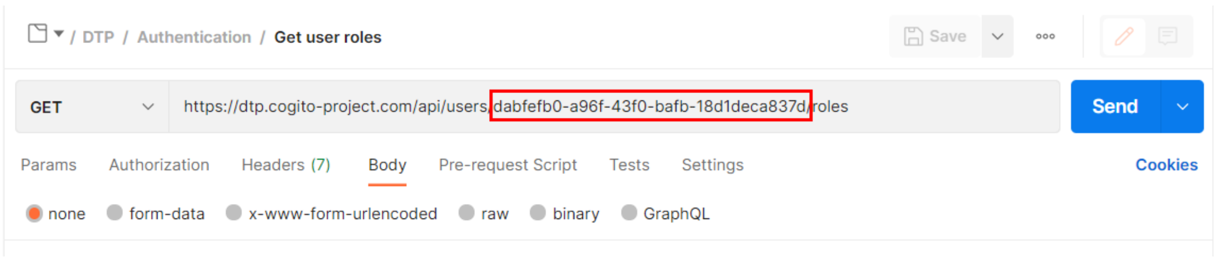

/ DTP / Authentication / POST User Info Request

POST https://auth.cogito-project.com/auth/realms/cogito/protocol/openid-connect/userinfo

Body: x-www-form-urlencoded

KEY	VALUE	DESCRIPTION
✓ client_id	cogito-digitar	
✓ username	[redacted]	
✓ password	[redacted]	
✓ client_secret	M98QPtcQXC23VXaCxvl8KOrcppdtlGCid_2	
✓ access_token	eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2l...	

Figure 28 - Request to retrieve user information using as a query parameter the access token


/ DTP / Authentication / Get user roles

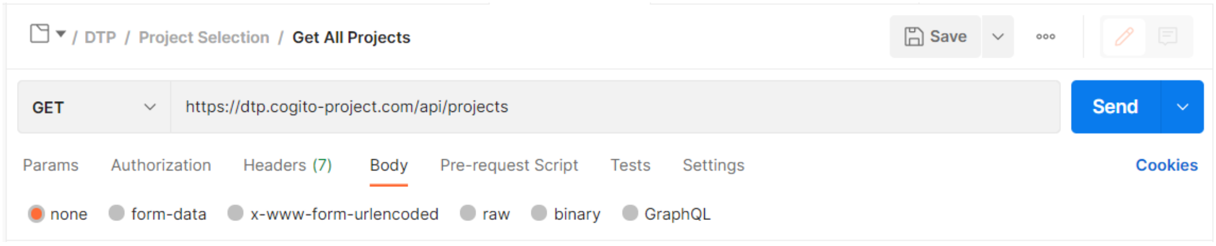
GET https://dtp.cogito-project.com/api/users/dabfefb0-a96f-43f0-bafb-18d1deca837d/roles

Body: none

Figure 29 - Request to retrieve the user roles. The user ID (in red box) is used to create the API endpoint.

2.3.3.2 *Project List Request*

By making the query depicted in Figure 30, the project list is retrieved from the DT Platform. The response is parsed within the DigiTAR tool, and the names of the available projects are displayed in the corresponding menu of the DigiTAR tool (i.e., the menu with which the user selects a project). The project IDs are used for the BIM Model Request, as described in Section 2.3.3.3.


/ DTP / Project Selection / Get All Projects

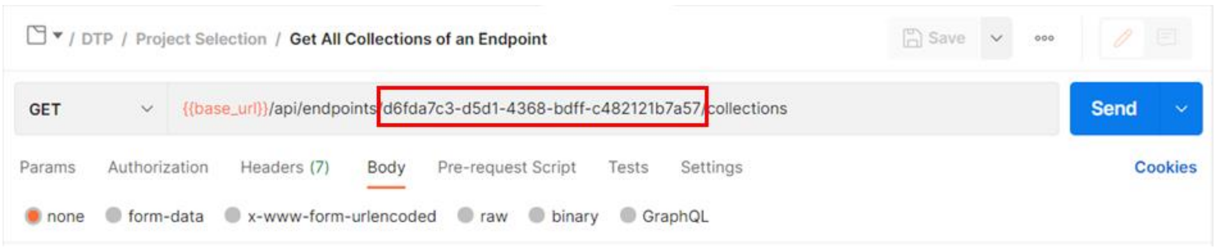
GET https://dtp.cogito-project.com/api/projects

Body: none

Figure 30 - Request to the DT Platform to retrieve the project list

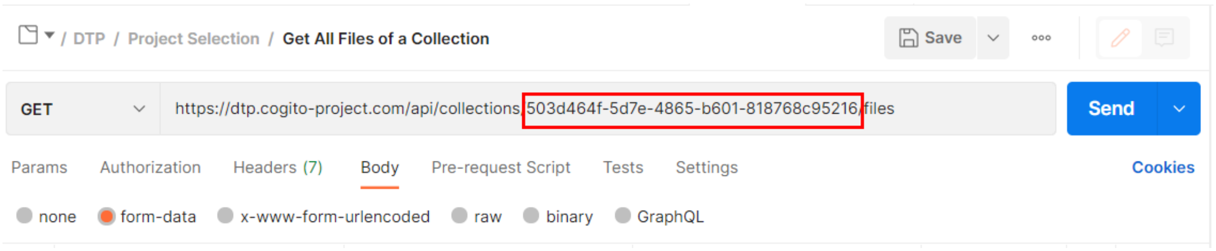
2.3.3.3 *BIM Model Request*

As mentioned before, there are dedicated endpoints for each interaction between the DigiTAR tool and the DT Platform. To request for the BIM model, the ID of the appropriate endpoint is used to create the API endpoint, as depicted in Figure 31. By retrieving the available collections for the specific endpoint using the query in Figure 31 and filtering the query response by a project ID, the collection ID for the specific project is acquired. This collection ID is subsequently used to query for the files that belong to this specific collection. As depicted in Figure 32, the collection ID is used to create the specific API endpoint. An example of this query's response is depicted in Figure 33. The files can be retrieved from the DT Platform using their "url" field. For the Safety mode, the IFC file enriched with the Safety information is requested and loaded respectively in the application.



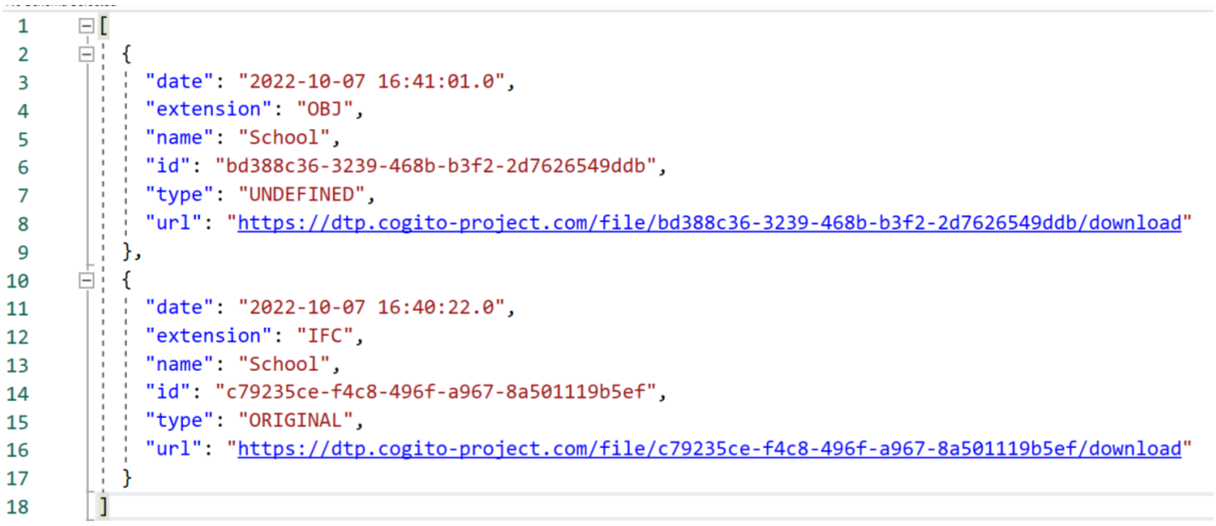
```
GET {{base_url}}/api/endpoints/d6fda7c3-d5d1-4368-bdff-c482121b7a57/collections
```

Figure 31 - Request to the DT Platform to retrieve the collections that are available for a specific endpoint. The endpoint ID (in red box) is used to create the API endpoint. Here, the endpoint ID corresponds to the endpoint that is dedicated to the BIM model request.



```
GET https://dtp.cogito-project.com/api/collections/503d464f-5d7e-4865-b601-818768c95216/files
```

Figure 32 – Request to the DT Platform to retrieve the files of a specific collection. The collection ID (in red box) is used to create the API endpoint.

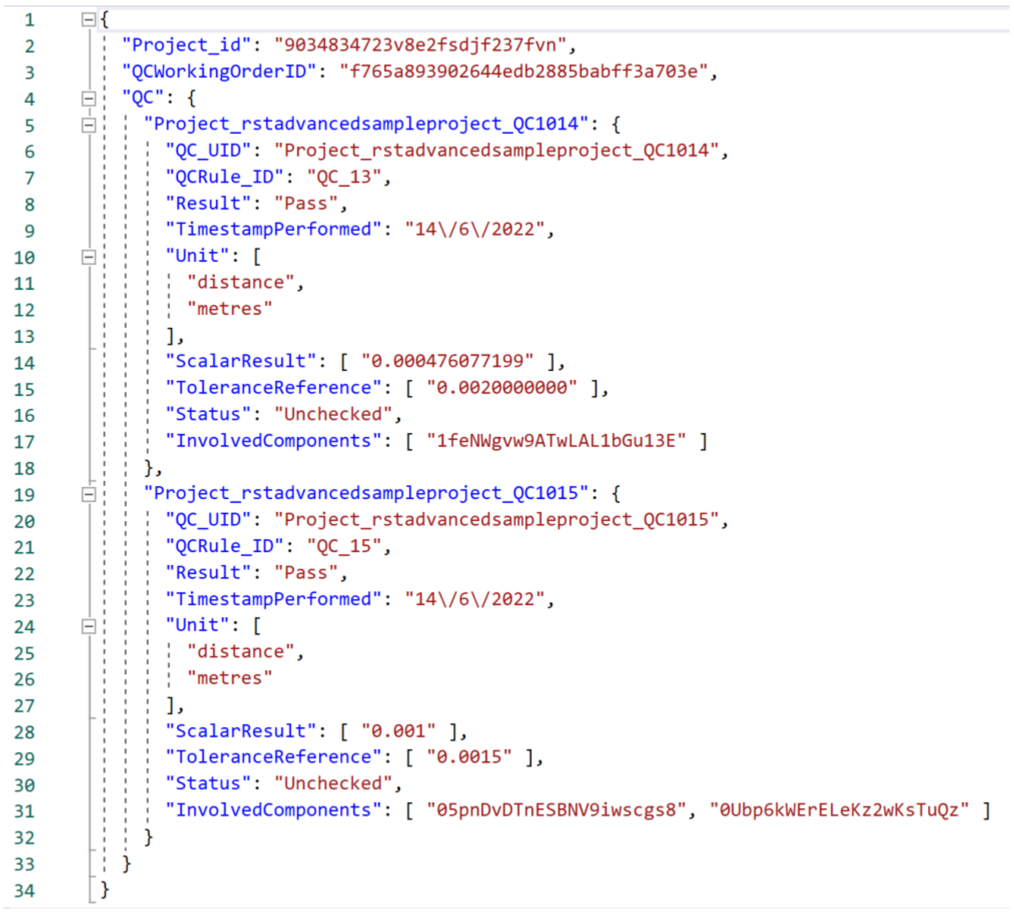


```json
[
  {
    "date": "2022-10-07 16:41:01.0",
    "extension": "OBJ",
    "name": "School",
    "id": "bd388c36-3239-468b-b3f2-2d7626549ddb",
    "type": "UNDEFINED",
    "url": "https://dtp.cogito-project.com/file/bd388c36-3239-468b-b3f2-2d7626549ddb/download"
  },
  {
    "date": "2022-10-07 16:40:22.0",
    "extension": "IFC",
    "name": "School",
    "id": "c79235ce-f4c8-496f-a967-8a501119b5ef",
    "type": "ORIGINAL",
    "url": "https://dtp.cogito-project.com/file/c79235ce-f4c8-496f-a967-8a501119b5ef/download"
  }
]
```

Figure 33 - The JSON file with the files that belong to a specific collection in the DT Platform

2.3.3.4 QC Results Request

In the Quality Control mode, the DigiTAR receives the QC results for the specific project. An example of the JSON file with the Geometric QC results is depicted in Figure 34. The properties received in the JSON body request are presented in Table 3. An example of the JSON file with the Visual QC results is depicted in Figure 35, while the properties received in the JSON body request are presented in Table 4.

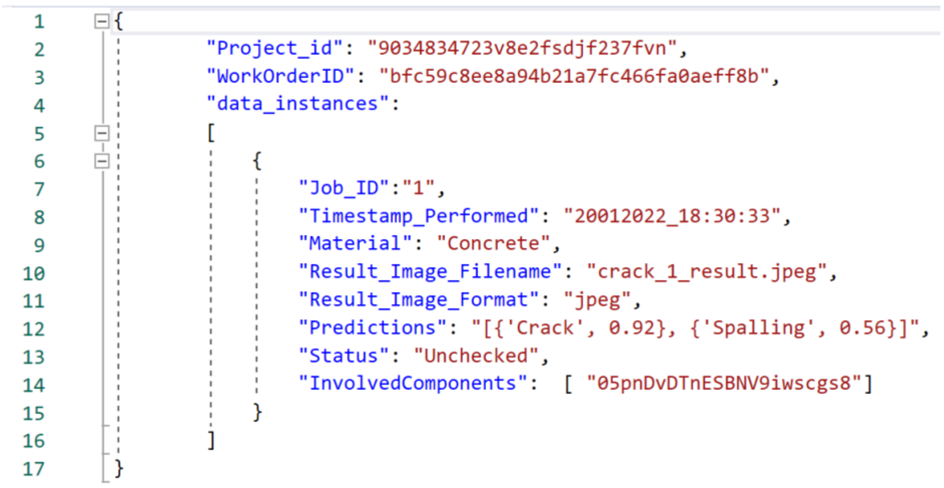


```
{
    "Project_id": "9034834723v8e2fsdjf237fvn",
    "QCWorkingOrderID": "f765a893902644edb2885babff3a703e",
    "QC": {
        "Project_rstadvancedsampleproject_QC1014": {
            "QC_UID": "Project_rstadvancedsampleproject_QC1014",
            "QCRule_ID": "QC_13",
            "Result": "Pass",
            "TimestampPerformed": "14\/6\/2022",
            "Unit": [
                "distance",
                "metres"
            ],
            "ScalarResult": [ "0.000476077199" ],
            "ToleranceReference": [ "0.0020000000" ],
            "Status": "Unchecked",
            "InvolvedComponents": [ "1feNWgvw9ATwLAL1bGu13E" ]
        },
        "Project_rstadvancedsampleproject_QC1015": {
            "QC_UID": "Project_rstadvancedsampleproject_QC1015",
            "QCRule_ID": "QC_15",
            "Result": "Pass",
            "TimestampPerformed": "14\/6\/2022",
            "Unit": [
                "distance",
                "metres"
            ],
            "ScalarResult": [ "0.001" ],
            "ToleranceReference": [ "0.0015" ],
            "Status": "Unchecked",
            "InvolvedComponents": [ "05pnDvDTnESBNV9iwscgs8", "0Ubp6kWErELeKz2wKsTuQz" ]
        }
    }
}
```

Figure 34 - The JSON file with the Geometric QC results received by DigiTAR in Quality Control mode

Table 3: Geometric QC results properties

Property	Description
Project_id	The project ID
QCWorkingOrderID	The QC workorder ID
QC_UID	The unique ID of the QC
QCRule_ID	The QC rule ID
Result	The QC result ("Pass"or "Fail")
TimestampPerformed	The timestamp when the QC was actually performed
Unit	The unit measurement
ScalarResult	The scalar result of the QC test
ToleranceReference	The tolerance of the QC test
Status	The status of the QC test
InvolvedComponents	The unique IFC identifiers of the components involved in the QC result



```json
{
        "Project_id": "9034834723v8e2fsdjf237fvn",
        "WorkOrderID": "bfc59c8ee8a94b21a7fc466fa0aeff8b",
        "data_instances":
        [
            {
                "Job_ID":"1",
                "Timestamp_Performed": "20012022_18:30:33",
                "Material": "Concrete",
                "Result_Image_Filename": "crack_1_result.jpeg",
                "Result_Image_Format": "jpeg",
                "Predictions": "[{'Crack', 0.92}, {'Spalling', 0.56}]",
                "Status": "Unchecked",
                "InvolvedComponents":  [ "05pnDvDTnESBNV9iwscgs8"]
            }
        ]
}
```

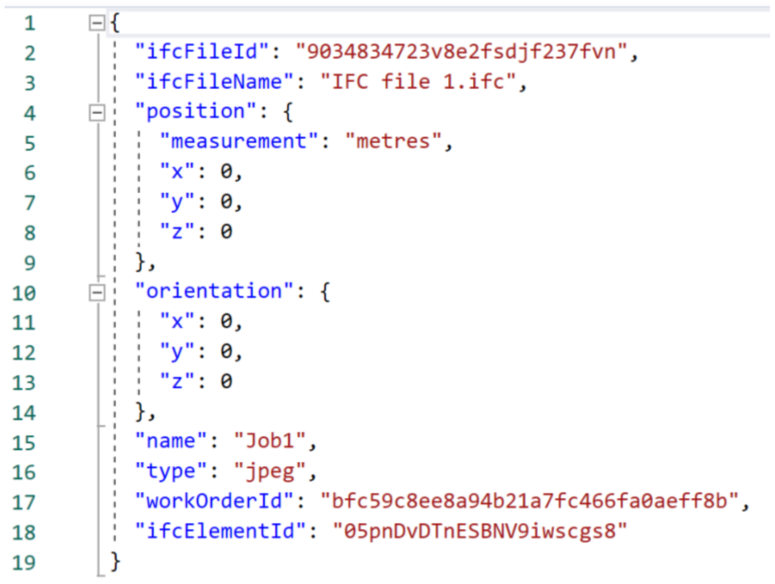
Figure 35 - The JSON file with the Visual QC results received by DigiTAR in Quality Control mode

Table 4: Visual QC results properties

Property	Description
Project_id	The project ID
WorkOrderID	The QC workorder ID
Job_ID	The QC job ID
Timestamp_Performed	The timestamp when the QC was actually performed
Material	The material of the involved component (Concrete, Steel or Undefined)
Result_Image_Filename	The filename of the tested image
Result_Image_Format	The format of the tested image
Predictions	The Visual QC predicted labels and probabilities
Status	The status of the QC test
InvolvedComponents	The unique IFC identifiers of the components involved in the QC result

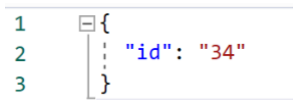
2.3.3.5 Pre-processing Mode Requests

Creation of the pre-processing job request: In the Pre-processing mode, DigiTAR performs as a data acquisition tool. When the user of the DigiTAR application creates a pre-processing job, a request is made to the endpoint provided by the Visual Data Pre-processing module with the properties of the job, as depicted in the JSON in Figure 36. As can be seen, the local position and orientation of the HoloLens camera (i.e., the user's position and orientation) are included in the metadata. The properties that should be passed in the JSON body request are presented in Table 5. In its response to this query, the Visual Data Pre-processing module includes the job ID, i.e., the ID that it has assigned to the newly created pre-processing job. A sample JSON with the Visual Data Pre-processing module's response is depicted in Figure 37.



```json
{
  "ifcFileId": "9034834723v8e2fsdjf237fvn",
  "ifcFileName": "IFC file 1.ifc",
  "position": {
    "measurement": "metres",
    "x": 0,
    "y": 0,
    "z": 0
  },
  "orientation": {
    "x": 0,
    "y": 0,
    "z": 0
  },
  "name": "Job1",
  "type": "jpeg",
  "workOrderId": "bfc59c8ee8a94b21a7fc466fa0aeff8b",
  "ifcElementId": "05pnDvDTnESBNV9iwscgs8"
}
```

Figure 36 - The JSON file defining the pre-processing job generated within DigiTAR in for the Pre-processing mode



```json
{
  "id": "34"
}
```

Figure 37 - The JSON file, which is the response of the Visual Data Pre-processing module to the request depicted in Figure 36, includes the ID of the newly created pre-processing job.

Table 5 – Pre-processing job properties

Property	Description
ifcFileId	The unique identifier for the IFC file
ifcFileName	The name of the IFC file
position	The position of the device
orientation	The orientation of the device
name	The name of the device
type	The image type (e.g., .png, .jpeg)
workOrderId	The workorder ID
ifcElementId	The unique IFC identifier of the component involved in the pre-processing job

The position property has the sub-properties illustrated in Table 6.

Table 6 - Sub-properties of position

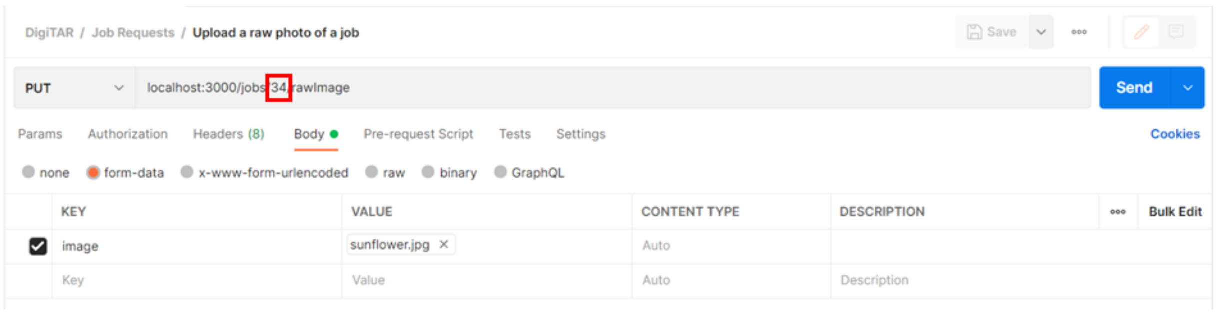
Position property	Description
measurement	The measurement that will be used for calculating the position of the device. *Metres* or *Centimetres*
x	The x coordinate of the device's position
y	The y coordinate of the device's position
z	The z coordinate of the device's position

The orientation property has the sub-properties illustrated in Table 7.

Table 7 - Sub-properties of orientation

Orientation property	Description
x	The x axis rotation of the device
y	The y axis rotation of the device
z	The z axis rotation of the device

Upload raw photos request: The raw photos that are attached to each job are either captured by the user during the runtime of the DigiTAR application or selected by the HoloLens' gallery. The raw images are passed to the Visual Data Pre-processing module using the API endpoint depicted in Figure 38. As can be seen, the pre-processing job ID is included in the endpoint's URL. The response to this request is the filename that the Visual Data Pre-processing module has assigned to the uploaded photo, as depicted in Figure 39.


DigiTAR / Job Requests / **Upload a raw photo of a job**

PUT localhost:3000/jobs/34/rawImage

	KEY	VALUE	CONTENT TYPE	DESCRIPTION
✓	image	sunflower.jpg	Auto	
	Key	Value	Auto	Description

Figure 38 - API endpoint for uploading a raw photo linked to a particular job

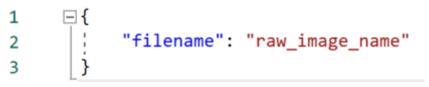
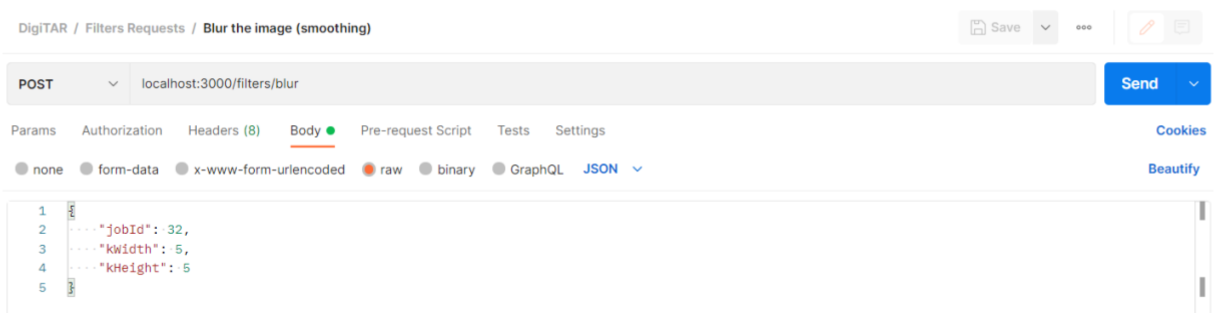

```
{
    "filename": "raw_image_name"
}
```

Figure 39 - The JSON file, which is the response of the Visual Data Pre-processing module to the request depicted in Figure 38, includes the filename of the newly uploaded raw photo.

Apply filter request: When the user selects a filter to be applied to the image from the GUI of DigiTAR, the corresponding request is performed to the API endpoint of the Visual Data Pre-processing module. In the following, examples of filtering requests are provided (i.e., blurring, resizing, cropping, and changing brightness/contrast).

Example request for blurring an image: The API endpoint for blurring an image is depicted in Figure 40. A successful request returns HTTP 200 Status and the processed image. The properties that should be passed in the JSON body request are presented in Table 8.


DigiTAR / Filters Requests / **Blur the image (smoothing)**

POST localhost:3000/filters/blur

```
{
    "jobId": 32,
    "kWidth": 5,
    "kHeight": 5
}
```

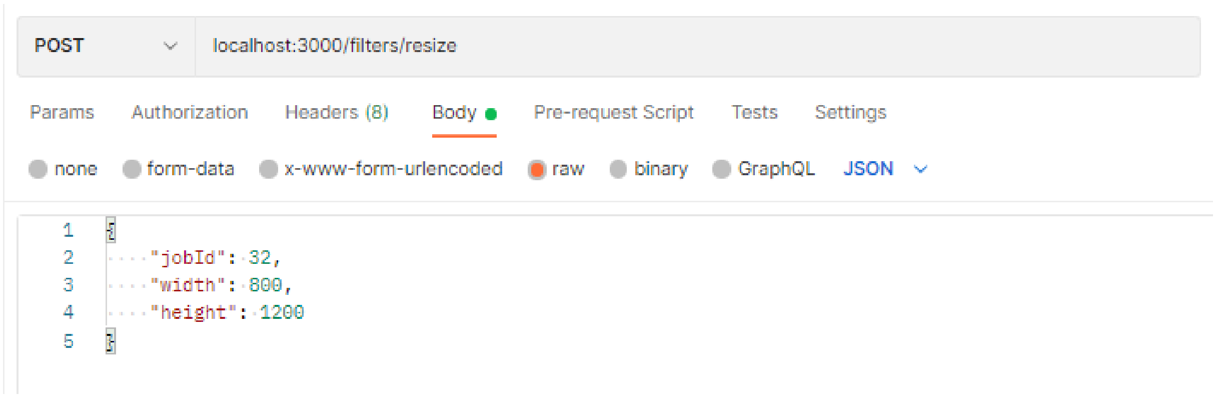
Figure 40 - API endpoint for requesting the blurring filter to be applied to the image

Table 8 - Blur properties

Property	Description

jobId	The unique identifier of the related job
kWidth	The width of the kernel used for Gaussian Blurring (should be odd)
kHeight	The height of the kernel used for Gaussian Blurring (should be odd)

Example request for resizing an image: The API endpoint for resizing the image related to a job is depicted in Figure 41. A successful request returns HTTP 200 Status and the processed image. The properties that should be passed in the JSON body request are presented in Table 9.



localhost:3000/filters/resize

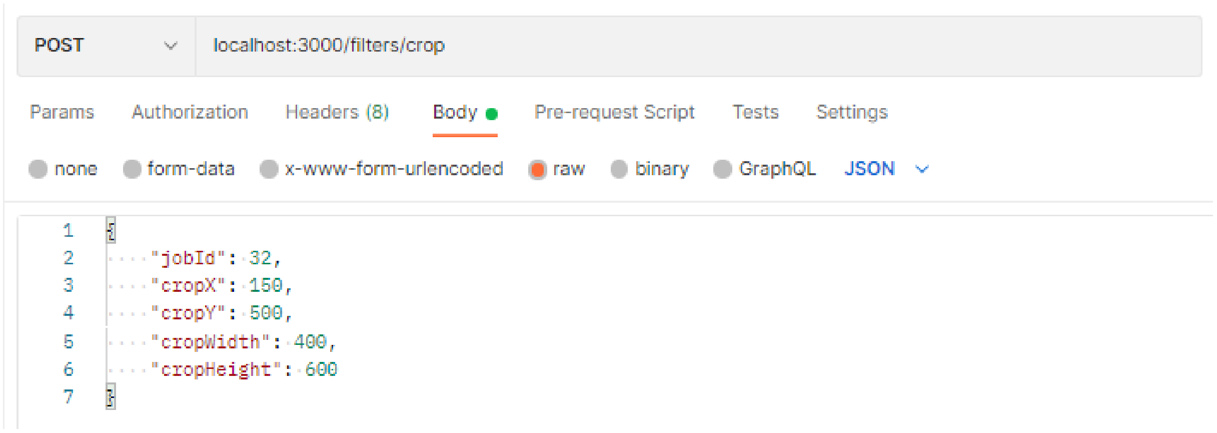
```
{
    "jobId": 32,
    "width": 800,
    "height": 1200
}
```

Figure 41 - API endpoint for resizing an image related to a job

Table 9 - Resize properties

Property	Description
jobId	The unique identifier of the related job
width	The desirable width of the processed image
height	The desirable height of the processed image

Example request for cropping an image: The API endpoint for cropping the image related to a job is depicted in Figure 42. A successful request returns HTTP 200 Status and the processed image. The properties that should be passed in the JSON body request are presented in Table 10.



localhost:3000/filters/crop

```
{
    "cropX": 150,
    "jobId": 32,
    "cropY": 500,
    "cropWidth": 400,
    "cropHeight": 600
}
```

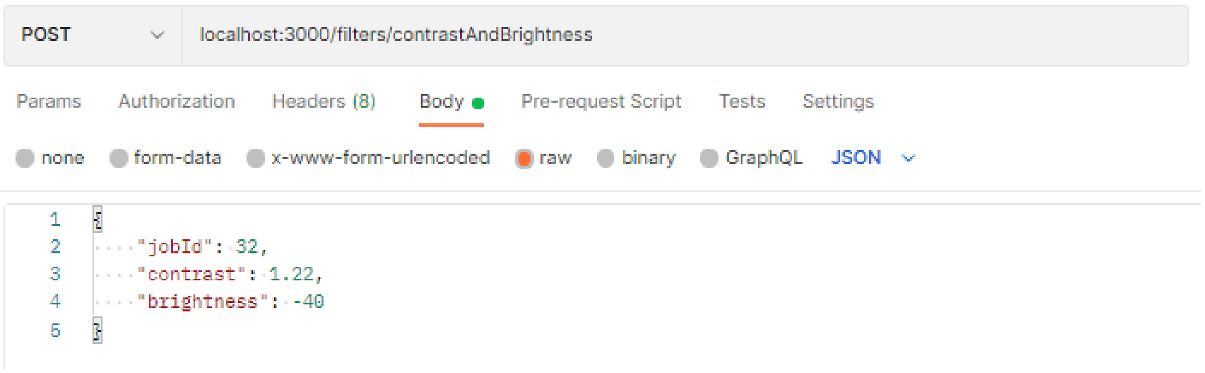
Figure 42 - API endpoint for cropping an image related to a job

Table 10 - Cropping properties

Property	Description
jobId	The unique identifier of the related job

cropX	X position of the starting cropping point (upper left point)
cropY	Y position of the starting cropping point (upper left point)
cropWidth	The desirable cropped width
cropHeight	The desirable cropped height

Example request for changing contrast and brightness of an image: The API endpoint for adjusting the contrast and brightness of the image [10] related to a job is depicted in Figure 43. A successful request returns HTTP 200 Status and the processed image. The properties that should be passed in the JSON body request are presented in Table 11.



```
localhost:3000/filters/contrastAndBrightness
```

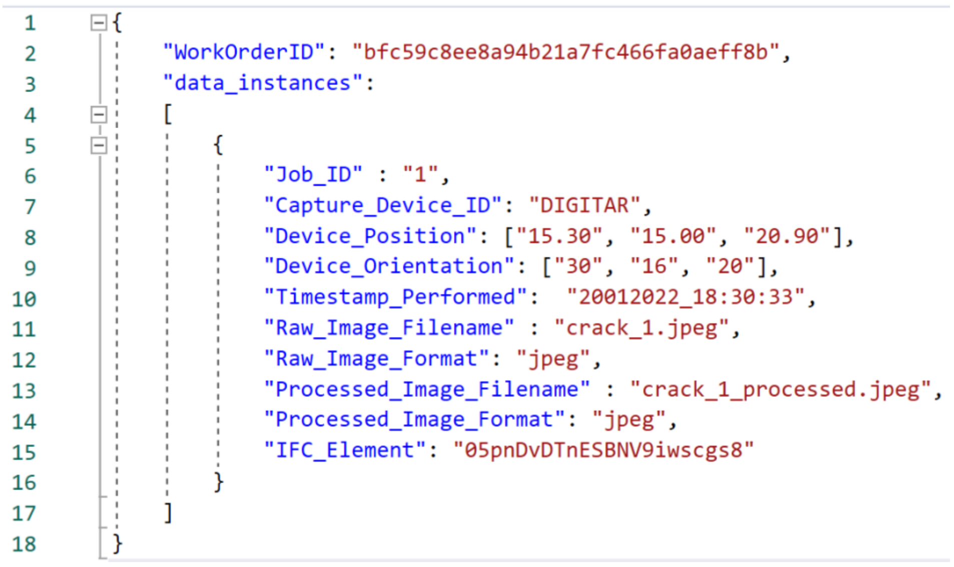
```
{
    "jobId": 32,
    "contrast": 1.22,
    "brightness": -40
}
```

Figure 43 - API endpoint for changing contrast and brightness of an image related to a job

Table 11 – Contrast and brightness properties

Property	Description
jobId	The unique identifier of the related job
contrast	The desirable contrast of the processed image (values from 0.30 – 2.00)
brightness	The desirable brightness of the processed image (values from -127 – 127)

Send confirmed pre-processed data and metadata to DT Platform: If the user of the DigiTAR tool selects to send the pre-processing job to the DT Platform, a request is made to the corresponding endpoint provided by the DT Platform. The properties included in the JSON file are depicted in Figure 44. The "Capture_Device_ID" property is set to a static value, i.e., "DIGITAR" for pre-processing jobs that are generated within DigiTAR.



```
{
    "WorkOrderID": "bfc59c8ee8a94b21a7fc466fa0aeff8b",
    "data_instances":
    [
        {
            "Job_ID" : "1",
            "Capture_Device_ID": "DIGITAR",
            "Device_Position": ["15.30", "15.00", "20.90"],
            "Device_Orientation": ["30", "16", "20"],
            "Timestamp_Performed":  "20012022_18:30:33",
            "Raw_Image_Filename" : "crack_1.jpeg",
            "Raw_Image_Format": "jpeg",
            "Processed_Image_Filename" : "crack_1_processed.jpeg",
            "Processed_Image_Format": "jpeg",
            "IFC_Element": "05pnDvDTnESBNV9iwscgs8"
        }
    ]
}
```

Figure 44 - The JSON file defining the pre-processing job properties that is sent from DigiTAR to the DT Platform

2.4 Usage Walkthrough

In this section, a brief introduction to some basic actions and gestures that HoloLens recognizes is firstly enclosed followed by a detailed walkthrough guide describing the usage of the DigiTAR tool and safety considerations to be taken into account when using the DigiTAR tool.

2.4.1 Introduction to Basic HoloLens Actions and Gestures

The basic interactions of the user with HoloLens are:

- **Touch**: With HoloLens the user can reach out and touch holograms. When HoloLens sees the user's hand, a floating pointer (like a mouse pointer) appears near the tip of the user's index finger to help the user target elements (see Figure 45).
- **Hand ray**. To use a hand ray, the user holds their hand in front of them, with their palm facing away. A laser pointer (hand ray) appears (see Figure 46).

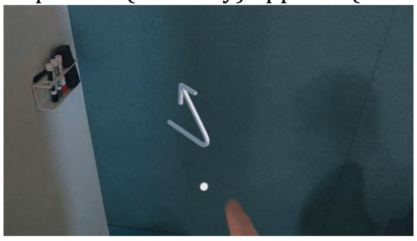


Figure 45 - Touch gesture on HoloLens

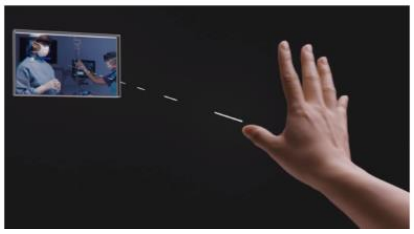
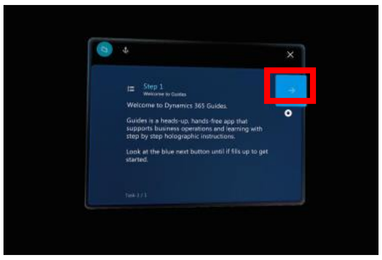


Figure 46 - Hand ray gesture

- **Gaze**: When the user gazes at an item, the pointer in Figure 47 corresponds to where the user's eyes are looking at. Often, the gaze is used together with the air tap. When the user gazes, they should turn their whole head, not just their eyes. The pointer will follow the movement of the user's gaze.
- **Air tap**: For the air tap gesture, the user holds their hand straight out in front of them in a loose fist, point their index finger straight up toward the ceiling, tap their finger down, and then quickly raise it back up again, as depicted in Figure 48.



Step 1
Welcome to Guides
Welcome to Dynamics 365 Guides.

Guides is a heads-up, hands-free app that supports business operations and learning with step by step holographic instructions.

Look at the blue next button until it fills up to get started.

Task 1 / 1

Figure 47 - Gaze on HoloLens

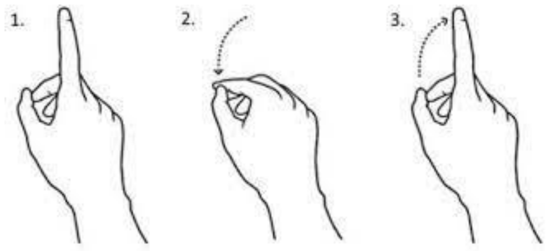


Figure 48 - Air tap gesture on HoloLens

- **Hand-tracking frame**: HoloLens has sensors that can see a few feet to either side of the user. When the user uses their hands, they need to keep them inside that frame, or HoloLens won't see them. However, the frame moves with the user as the user moves around. The hand tracking frame on HoloLens is depicted in Figure 49.

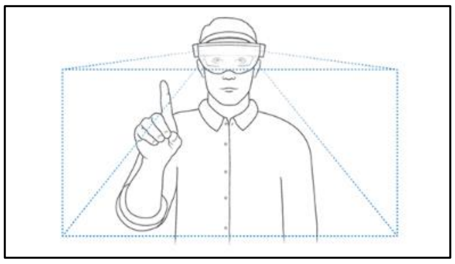


Figure 49 - Hand tracking frame on HoloLens

- **HoloLens2 gestures to open the Start menu:** The Start menu of HoloLens is depicted in Figure 50. The Start menu on HoloLens is where the user can open apps, see important status info, and access tools like the camera.

  There are two ways to open the Start Menu (see Figure 51):

  1. **Open the Start menu with two hands**
     - The user should hold out one of their hands with the palm facing up and look at their wrist. They should see a holographic Windows logo.
     - With the index finger of their other hand, they should touch the Windows logo.
  2. **Open the Start menu with one hand**
     - The user should hold out one of their hands with the palm facing up and look at their wrist. They should see a holographic Microsoft Windows logo.
     - With the hand that they are holding out, they should touch their index finger to their thumb in a pinching motion.

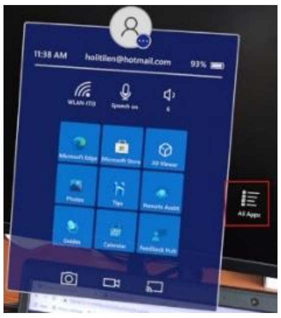


Figure 50 - Start menu on HoloLens

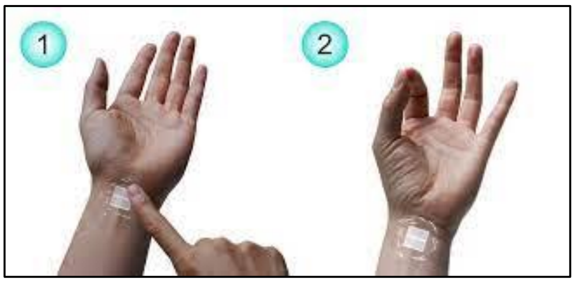


Figure 51 - Gesture to open the Start menu in HoloLens with (1) both hands, (2) one hand

2.4.2 Usage Walkthrough for the DigiTAR Application

Open the DigiTAR application: To open the application, the user should:

1. Perform the Start gesture to open the Start menu of HoloLens.
2. Use the Touch gesture to select **"All apps"** to the right (see Figure 52).
3. From the list of applications, select DigiTAR using the Touch gesture.

The first time the application is opened, the user's permission is asked to let DigiTAR access the HoloLens camera and microphone. The user should accept these permissions since they are necessary for the application to capture images/videos and use speech commands.

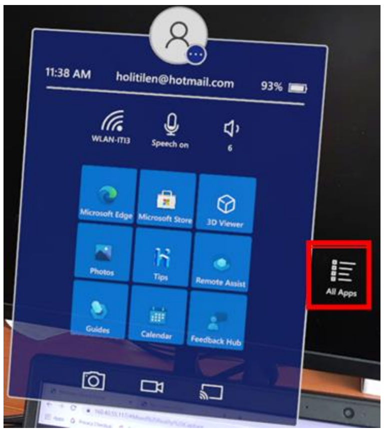


Figure 52 - Select "All apps" to see all applications installed on HoloLens

Close the DigiTAR application: To close the application (at any time), the user can perform the Start gesture (depicted in Figure 51) to open the Start menu of HoloLens and select the "Home" button (see Figure 53). Subsequently, the user should close the 2D box of the application (see Figure 54) to completely terminate the application. It is important to always make sure to close the 2D box before opening the application again, to avoid messing the calibration of the 3D BIM model and malfunctioning of the application.

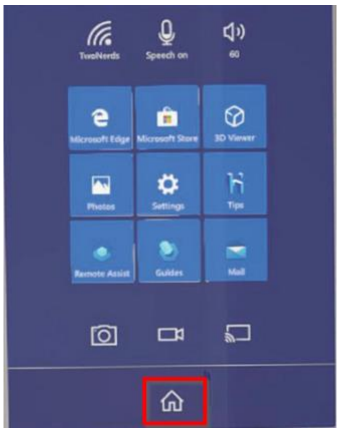


Figure 53 - To close the application, the user can select the "Home" button of the Start Menu

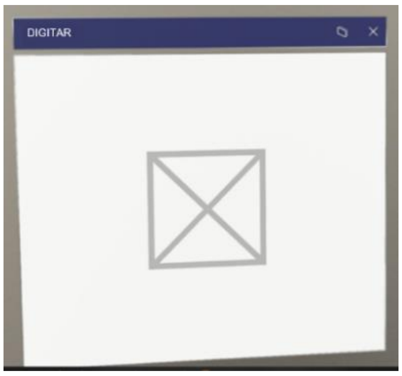


Figure 54 - The user should make sure to also close the 2D box of the application in order to completely terminate the application

Login menu: When the application is opened, the user is prompted to provide their credentials to login, as illustrated in Figure 55. To insert their username, the user can gaze and perform the air tap gesture on the username input field and the keyboard of HoloLens opens-up, as depicted in Figure 56. The user can type on the Hololen's keyboard using the Touch gesture. To use dictation, the user can air tap on the dictation button (in red in Figure 56) to activate it. After inserting their credentials, the user can air tap on the "Login" button in the *Login* menu (see Figure 55) to proceed with the authentication. If the user's credentials are validated, a welcome message is displayed to the user as depicted in Figure 57. Otherwise, the user is prompted to repeat the authentication procedure as depicted in Figure 58. To view and inspect the inserted password, the user can air tap on the "eye" button that is positioned at the right of the password's input field, as depicted in Figure 58. The user can also choose the "Exit" button to close the *Login* menu without proceeding with the application.

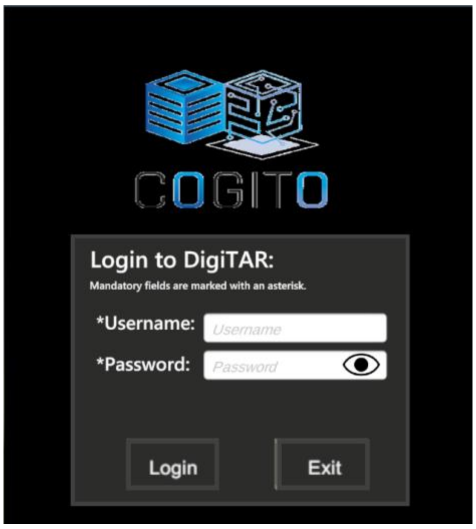


Figure 55 - DigiTAR tool user's login menu

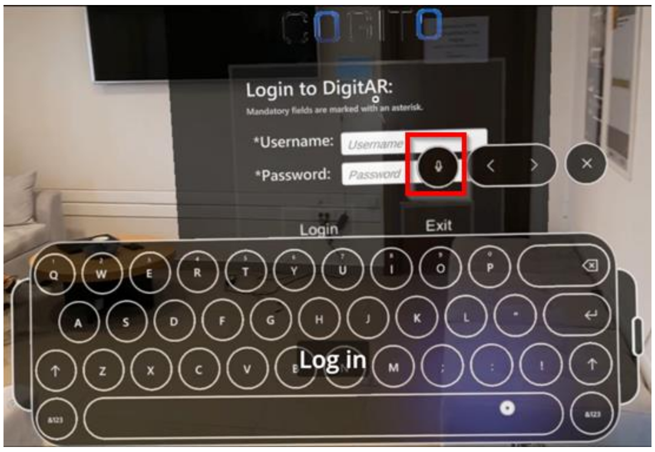


Figure 56 - When the user gazes and performs the air tap gesture on the input fields, the keyboard of HoloLens opens-up. User can also use dictation instead of typing.

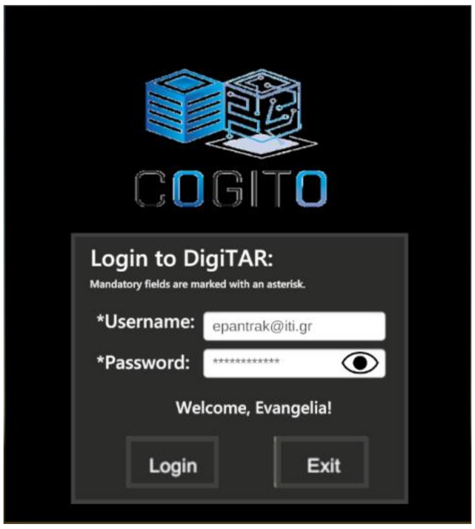


Figure 57 - If the credentials are validated, a welcome message is displayed to the user

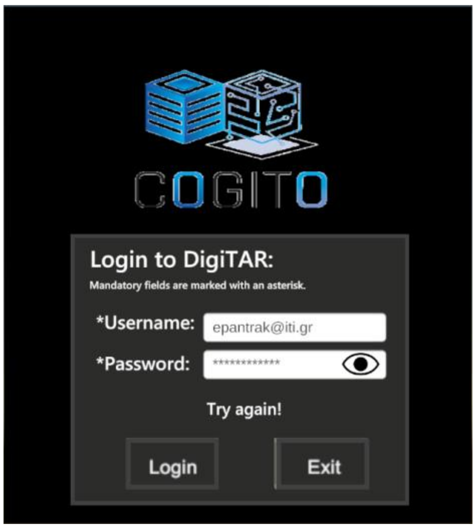

COGITO
Login to DigiTAR:
Mandatory fields are marked with an asterisk.
*Username: epantrak@iti.gr
*Password: ************
Try again!
Login
Exit

Figure 58 - If the credentials are not validated, the user is asked to retry the authentication process

Speech commands: The user can select the buttons in DigiTAR by gazing at them and performing the air tap gesture. Most buttons in DigiTAR can be alternatively selected with speech commands, without having to use hand gestures. The user is notified for the speech command that activates a button, by gazing at it. This way, a tooltip appears with the corresponding speech command, as depicted in Figure 59.

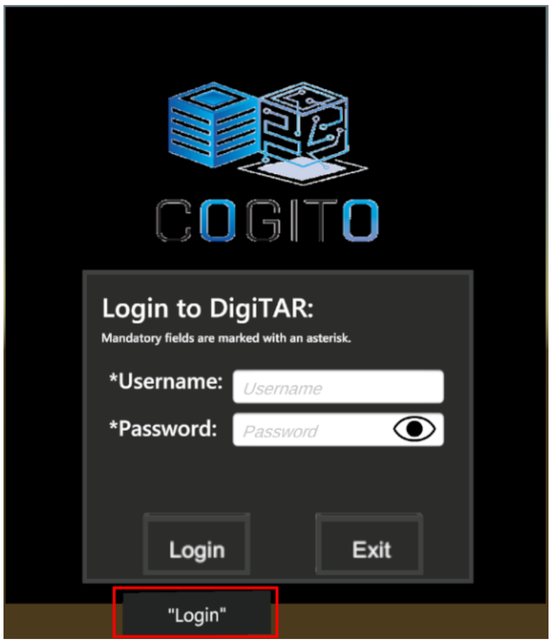


Figure 59 - When the user gazes at a button that can be activated with speech command, a 2D tooltip box appears with the corresponding speech command

Pin/Unpin menus: By default, all displayed menus in DigiTAR follow the user's movement. By using speech command "Pin", the currently displayed menu is pinned in its current position and stops following the user's movement. To restore the movement of the currently displayed menu, the user scan use speech command "Unpin".

Select project: After successful authentication, the menu in Figure 60 opens-up so that the user can select a project. The user can perform the air tap gesture on the input field so that a drop-down list with the available projects appears, as depicted in Figure 61. The project list is retrieved by a query to the DT

Platform, as described in Section 2.3.3.2. The user can select a project from the drop-down list with the air tap gesture.

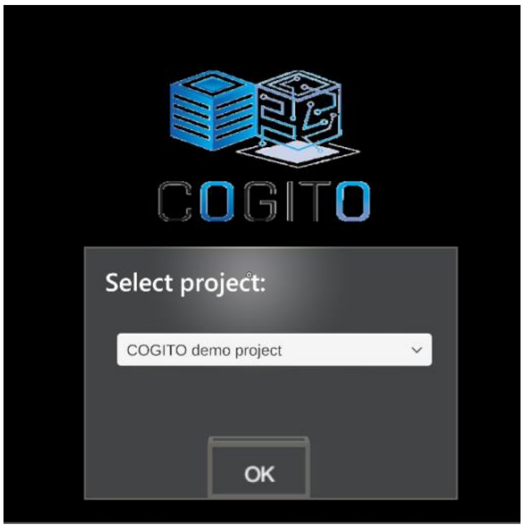


Figure 60 - The menu of DigiTAR tool for the user to select a project

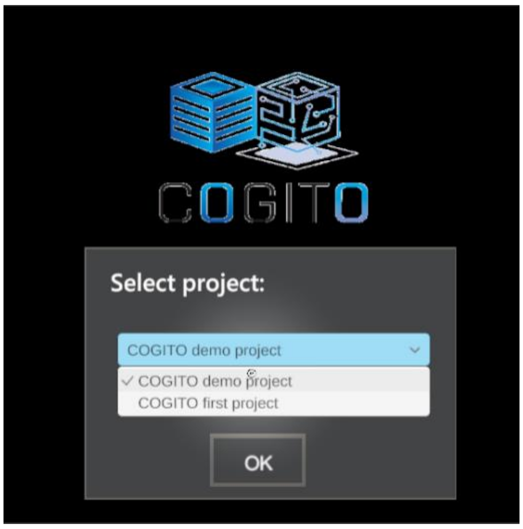


Figure 61 - The user can air tap to select a project from the drop-down list

Select mode: After selecting a project, the user is prompted to the *Select mode* menu of the DigiTAR tool, which is depicted in Figure 62. As already mentioned, there are three modes for the DigiTAR: the Pre-processing, the Quality Control and the Safety Control modes. The user can air tap on the corresponding button to select a mode. Alternatively, the user can select a mode by gazing at the desired button and use the corresponding speech command. The speech commands assigned to the buttons are "Pre-processing", "Quality" and "Safety", respectively. To go back to the *Select project* menu, the user can air tap on the "Back" button at the bottom of the menu. To close the menu without proceeding with the mode selection, the user can air tap on the "Exit" button.

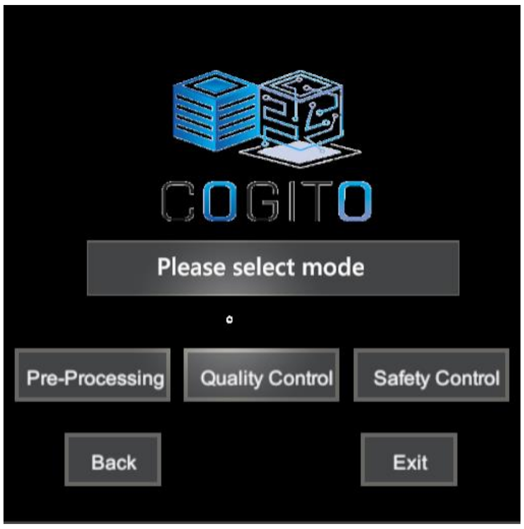


Figure 62 - The menu of the DigiTAR tool that allows the user to select a mode for the DigiTAR tool

Pre-processing mode: If the user selects the "Pre-Processing" button in the *Select mode* menu, the menu in Figure 63 opens-up. The user can air tap on the "Download BIM model" button to download the BIM model from the DT Platform or air tap on the "Visualize BIM model" button to load the most recently downloaded BIM model from the application's internal storage on the HoloLens. By selecting the first option, i.e., to download the BIM model from the DT Platform, the files are retrieved from the endpoints provided by the DT Platform, as described in the Section 2.3.3.3.

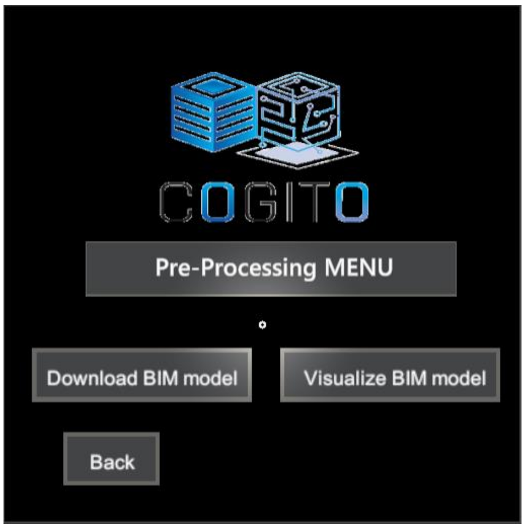


Figure 63 - For the Pre-processing mode, the user should firstly load the BIM model of the project

3D BIM model registration: The position of the image target in the 3D space for each IFC file will be known in advance to the user. To perform registration, the user can use speech command **"Scan"** to initialize the image target detection process. Afterwards, the user is expected to approach and look at the printed image target in the real world. After the image target has been detected, the IFC model will be visualised to be aligned to the real world. To maintain the registration for subsequent sessions, the user can use speech command **"Anchor"** to set the aligned 3D model as a spatial anchor, so that it can be loaded the next time the application is initialized eliminating the need to repeat the registration process.

Pre-processing mode main menu: When the user selects the Pre-processing mode in the DigiTAR tool, the elements included in the Visual Quality Control work orders, are displayed in red to notify the user, as depicted in Figure 64. The main menu of the Pre-processing mode of the DigiTAR tool is displayed in Figure 65. The "Toggle BIM model" button toggles on and off the 3D BIM model visualisation. If the user selects this button while viewing the 3D BIM model, the BIM model visualisation is switched off and only the 3D elements included in the Visual QC work orders are displayed (i.e., only the elements that are highlighted in red). To switch on again the 3D BIM model visualisation, the user can select this button again. Finally, the "Add job" button enables the generation of a pre-processing job within the DigiTAR tool.

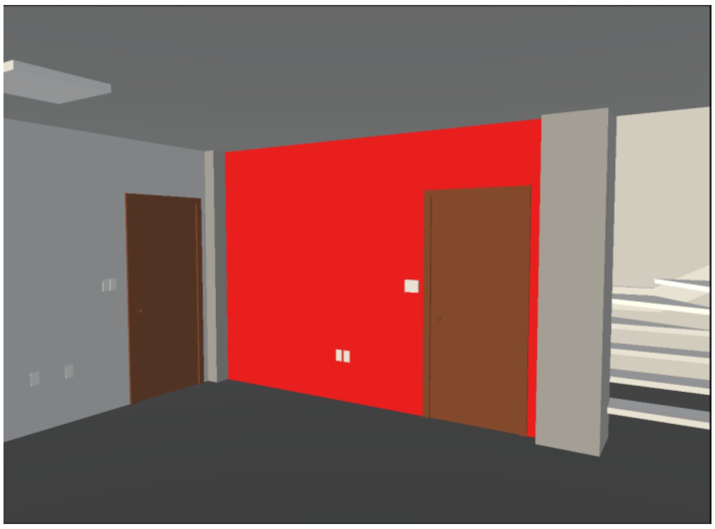


Figure 64 - The IFC elements included in Visual Quality Control work orders are highlighted in red

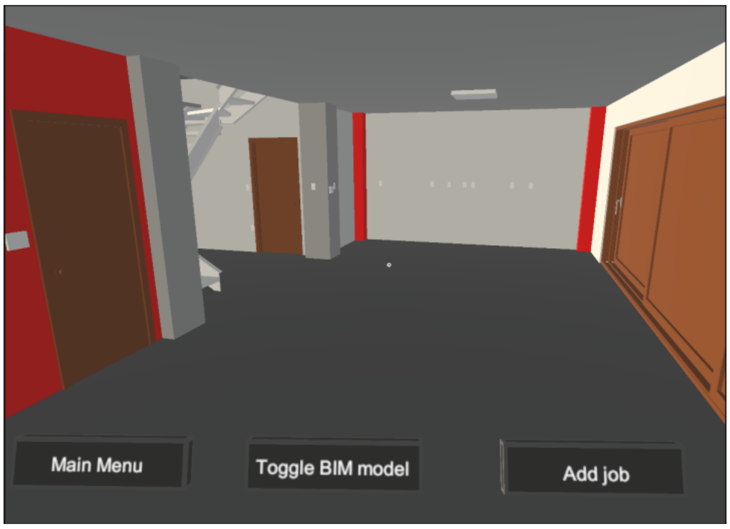


Figure 65 - The GUI displayed at the Pre-processing mode

Generate pre-processing job: The user can create a pre-processing job by selecting the corresponding "Add job" button from the displayed GUI of Pre-processing mode of the DigiTAR tool, which is displayed in Figure 65. After selecting the "Add job" button, the menu depicted in Figure 66, opens-up where the user is prompted to provide a name for the new pre-processing job. When the user performs the air tap gesture on the white input field, the keyboard of HoloLens opens-up and the user can type the desired name.

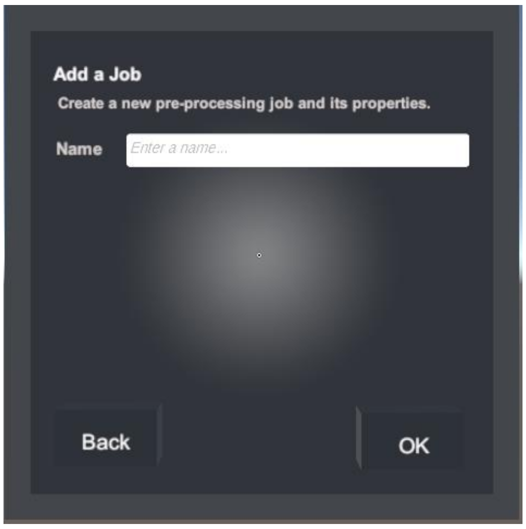


Figure 66 - Menu to add a pre-processing job

Attach 3D IFC elements to a pre-processing job: The user can gaze at a 3D IFC element and use speech command "Add" to attach this element to the newly generated job. The gazed 3D IFC element is highlighted in yellow to visually notify the user that the element has been selected. Subsequently, the menu depicted in Figure 67 opens-up that displays the name of the selected IFC element. The user can either change the IFC element that is associated to the job by selecting the "Change element" button or proceed by selecting the "OK" button.

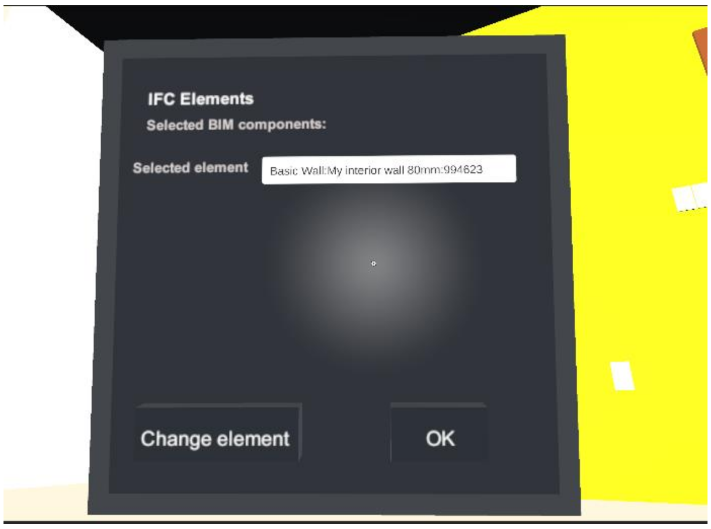


Figure 67 - The name of the selected IFC element (highlighted in yellow) that is attached to the pre-processing job is displayed

Link images to a pre-processing job: After an IFC element is attached to a pre-processing job, the menu in Figure 68 is displayed. Again, the user has the option to toggle between enabling and disabling the 3D BIM model visualisation using the “Toggle BIM model”. The user can capture an image to be linked to the job by selecting the “Capture image” button. Afterwards, the user can gaze at the desired location and use speech command “Capture” to capture an image that is linked to the job. The user can also attach images from the HoloLens gallery to the job by selecting the “Gallery” button. After the button has been selected, the pictures folder of HoloLens opens-up, as depicted in Figure 69. The user can navigate to the desired folder and select an image, as can be seen in Figure 70.

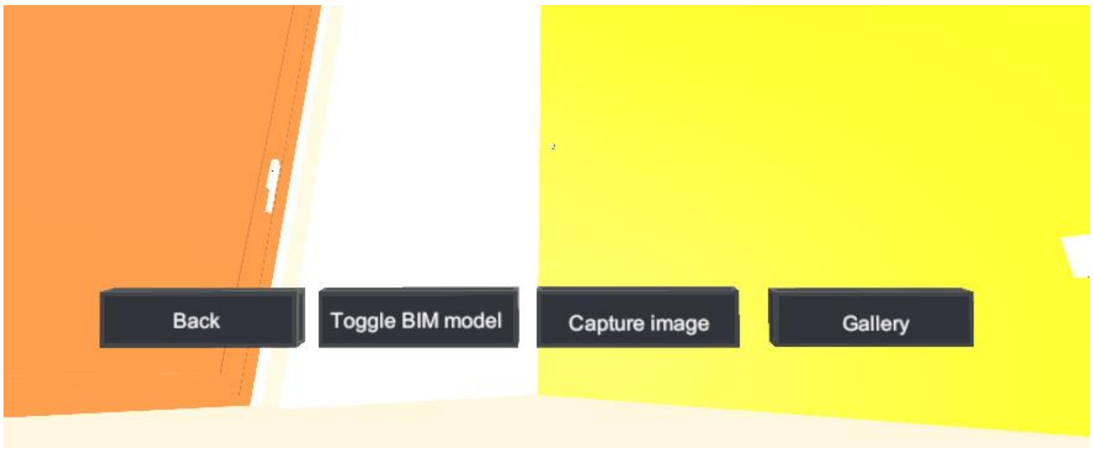


Figure 68 - The menu used for attaching images to a pre-processing job

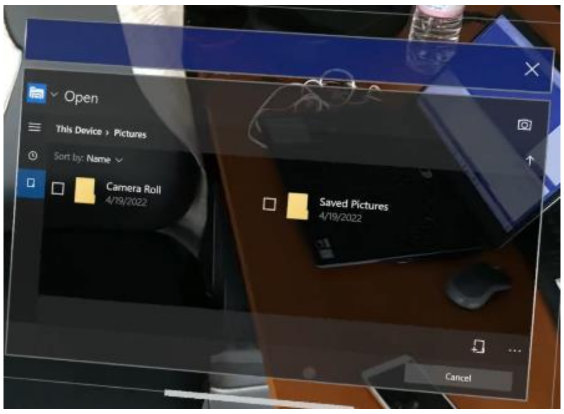


Figure 69 - The pictures folder on Hololens

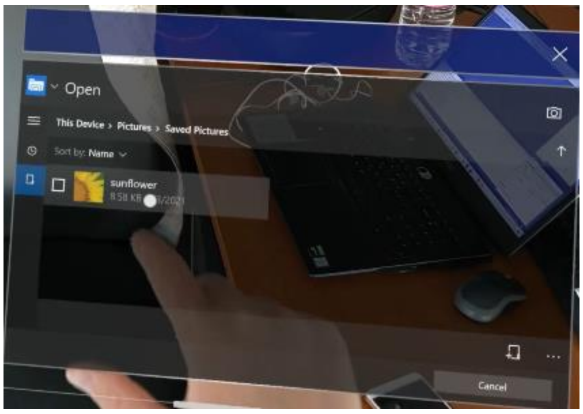


Figure 70 - The user can select an image to be attached to the pre-processing job

Preview menu: After linking an image to a pre-processing job (either by capturing an image at runtime or by selecting an image from the HoloLens gallery), the menu in Figure 71 opens-up that displays the attached image.

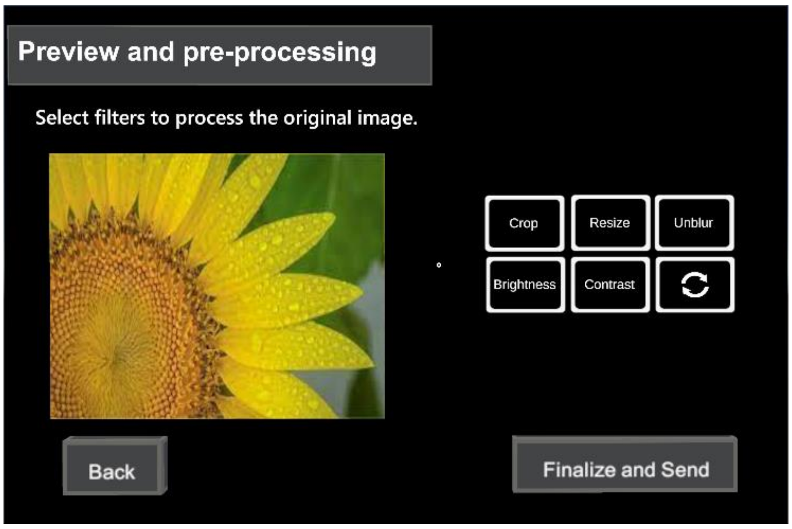

Preview and pre-processing
Select filters to process the original image.
Crop
Resize
Unblur
Brightness
Contrast
Back
Finalize and Send

Figure 71 - Menu for displaying the images attached to a pre-processing job

Apply filters and preview processed image: The user can select a filter from the menu displayed in Figure 71. For example, if the user selects the "Crop" filter, as displayed in Figure 72, the corresponding filter request is sent to the Visual Data Pre-processing module, using the API described in Section 2.3.3.5. Subsequently, the processed image is received from the Visual Data Pre-processing module and is displayed, replacing the original image that was displayed before. If the user selects the "Finalize and Send" button, the processed image and the pre-processing job are sent to the DT Platform so that they can be accessible for the Visual Quality Control module. After uploading to the DT Platform is completed, a confirmation message is depicted, as can be seen in Figure 73.

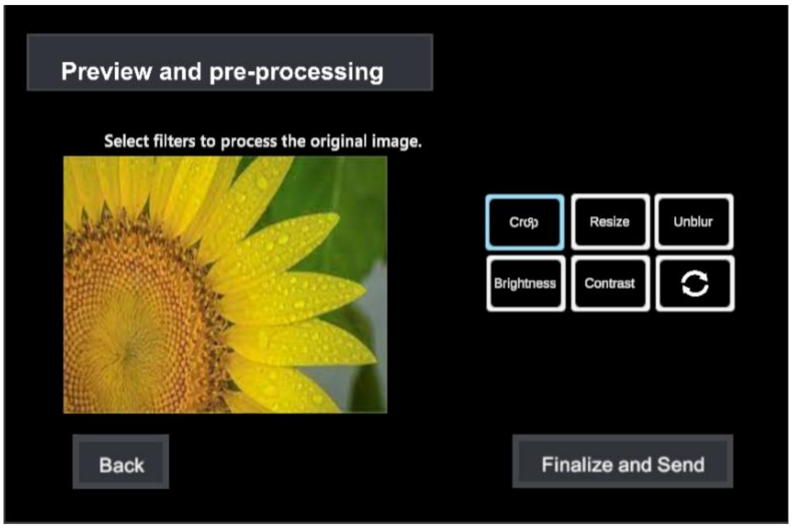

Preview and pre-processing
Select filters to process the original image.
Crop
Resize
Unblur
Brightness
Contrast
Back
Finalize and Send

Figure 72 - The user can select a filter by selecting the corresponding button

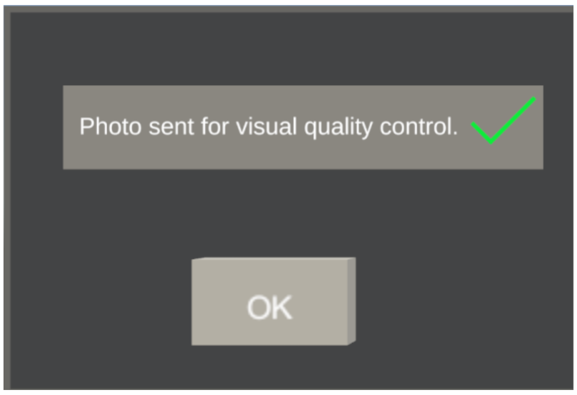


Figure 73 - Confirmation menu when processed image has been uploaded to the DT Platform and is accessible for the Visual Quality Control module

Quality Control mode: If the user selects the Quality Control mode in the *Select mode* menu in Figure 62, the DigiTAR tool downloads the Geometric and Visual Quality Control results for the specific project from the DT Platform using the API described in Section 2.3.3.4.

QC results notification: A 3D QC tag is attached to the 3D elements that are included in the Geometric and Visual QC results. To visually notify the user, the 3D tag is red for IFC elements that have a “Fail” Geometric QC result and blue for the IFC elements that have a “Pass” Geometric QC result. Similarly, for the Visual QC results, the 3D tag is red for IFC elements that have detected defects and blue for the IFC elements with no detected defects in the Visual QC result. A Geometric QC tag and a Visual QC tag are depicted in Figure 74. Examples illustrating how the 3D QC tags are pinned on the 3D IFC elements can be seen in Figure 75. Using speech command “Hide BIM”, the user can view only the 3D QC tags and not the 3D representations of the IFC elements, as depicted in Figure 76. Using speech command “Show BIM”, the user can switch back to viewing the 3D representations of the IFC elements. Alternatively, the user can view only the 3D representations of the IFC element that they are currently looking at, i.e., the 3D representation of each IFC element is rendered and visualised only when the user looks at the IFC element. During this visualisation, the outline of the object is highlighted to better illustrate that the 3D object is activated based on user’s gaze. An illustration of the latter visualisation method, which can be activated with speech command “Show outline”, is depicted Figure 77.

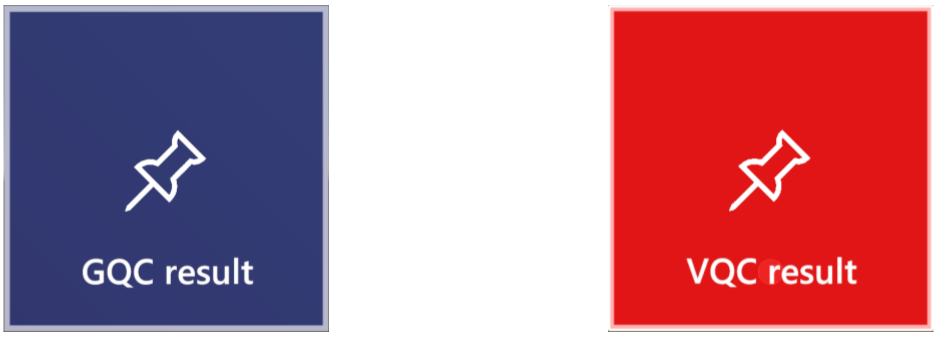


Figure 74 - The 3D QC tags that are pinned to the 3D elements that are included in Geometric QC (left) and Visual QC (right). Blue color indicates "Pass" Geometric QC result and "Non defect" Visual QC result. Red color indicates "Fail" Geometric QC result and detected defect in Visual QC result.

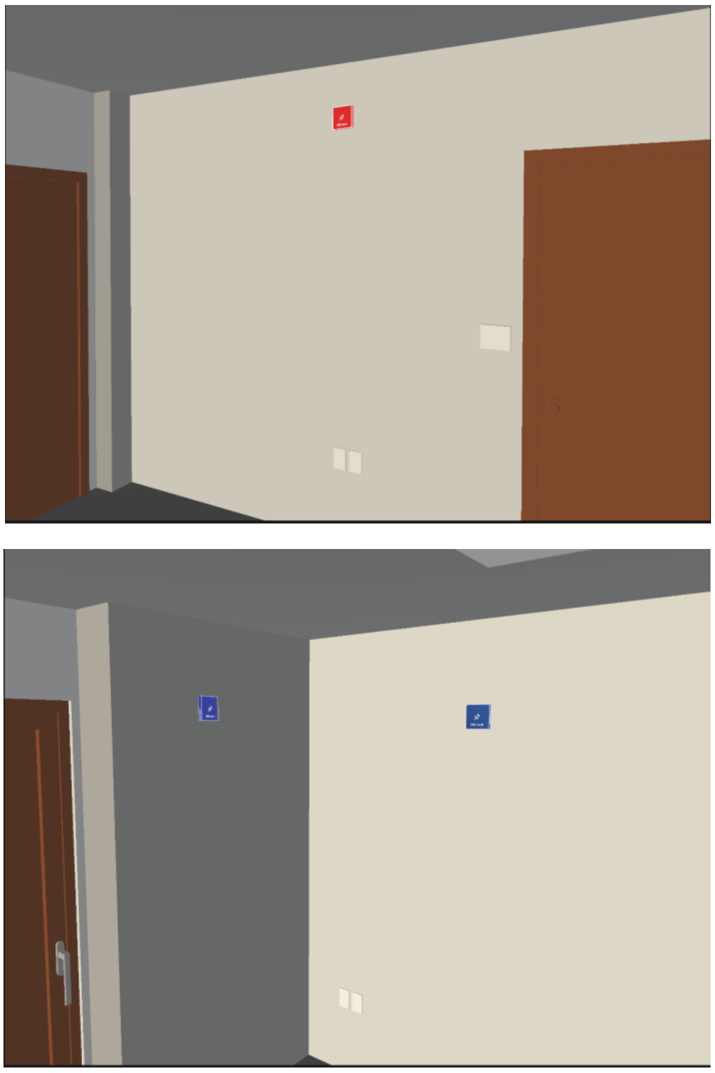


Figure 75 – Examples illustrating how the 3D QC tags are pinned to the elements included in the QC results

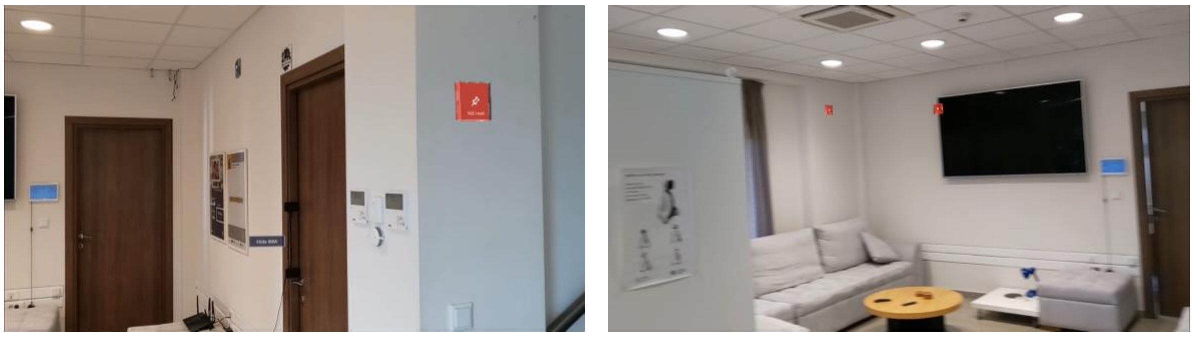


Figure 76 – The user can view only the 3D QC tags using speech command "Hide BIM"

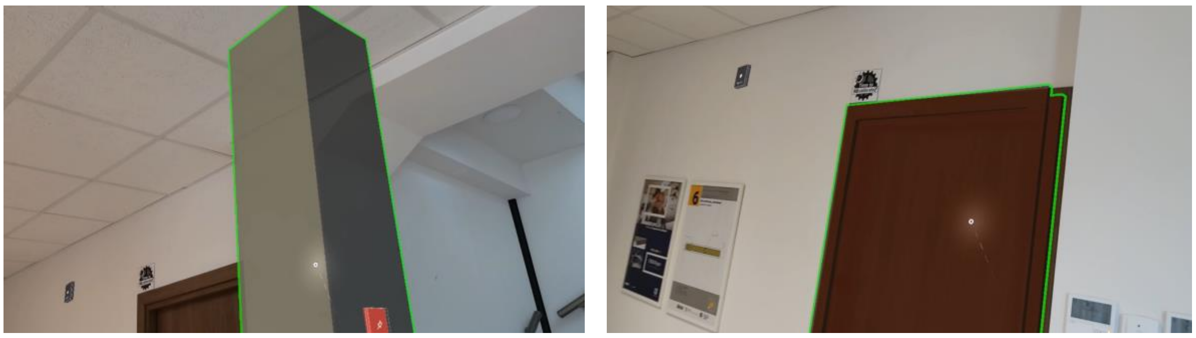
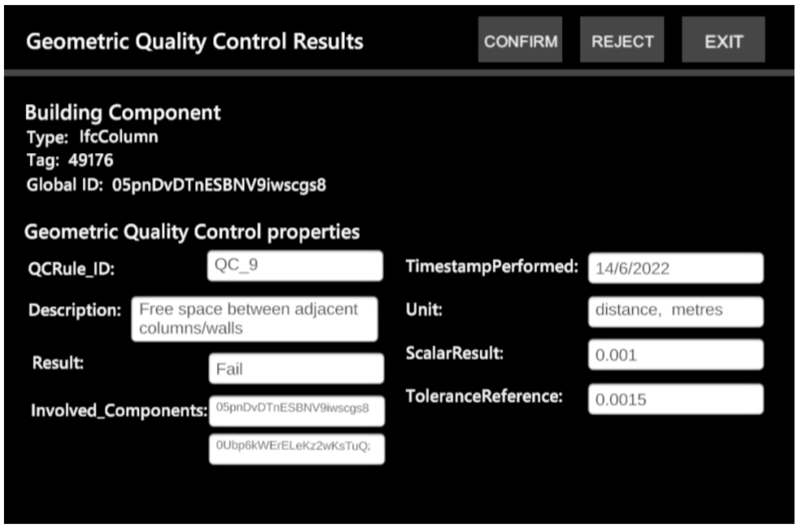


Figure 77 - The 3D representation of each IFC element is visualised only when the user looks at the IFC element. The outline of the object is highlighted to better illustrate that the 3D object is activated based on user's gaze.

QC results details visualisation: The user can air tap on the 3D QC tags to view details for the QC result. More specifically, the menu in Figure 78 is displayed for an example of a "Fail" Geometric QC result. The corresponding menu for an example of a "Pass" Geometric QC result is depicted in Figure 79. An example of a Visual QC result with defect detection is depicted in Figure 80. The user can select the "Show original image" button to view the original image (i.e., without the Visual QC annotations), as can be seen in Figure 81. The user can select the "Confirm" button or the "Reject" button to confirm or reject the QC result, respectively.

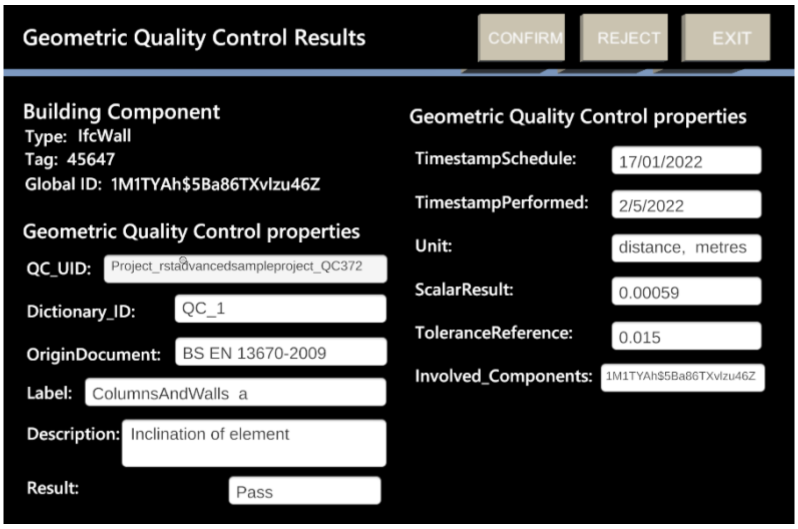


Building Component
Type: IfcColumn
Tag: 49176
Global ID: 05pnDvDTnESBNV9iwscgs8

Geometric Quality Control properties

QCRule_ID:	QC_9	TimestampPerformed:	14/6/2022
Description:	Free space between adjacent columns/walls	Unit:	distance, metres
Result:	Fail	ScalarResult:	0.001
Involved_Components:	05pnDvDTnESBNV9iwscgs8 0Ubp6kWErELeKz2wKsTuQ;	ToleranceReference:	0.0015

Figure 78 - Menu displaying a "Fail" QC result



Geometric Quality Control Results

Building Component
Type: IfcWall
Tag: 45647
Global ID: 1M1TYAh$5Ba86TXvlzu46Z

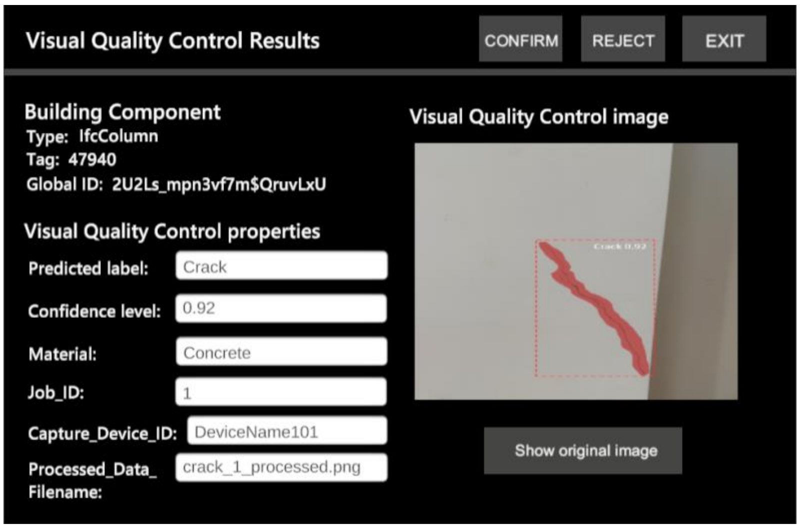
Geometric Quality Control properties

QC_UID:	Project_rstadvancedsampleproject_QC372
Dictionary_ID:	QC_1
OriginDocument:	BS EN 13670-2009
Label:	ColumnsAndWalls a
Description:	Inclination of element
Result:	Pass

Geometric Quality Control properties

TimestampSchedule:	17/01/2022
TimestampPerformed:	2/5/2022
Unit:	distance, metres
ScalarResult:	0.00059
ToleranceReference:	0.015
Involved_Components:	1M1TYAh$5Ba86TXvlzu46Z

Figure 79 - Menu displaying a "Pass" Geometric QC result



Visual Quality Control Results

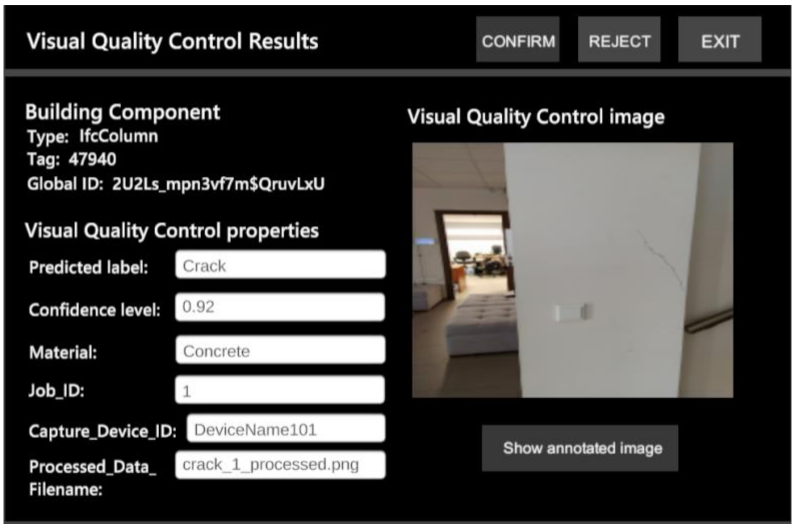
Building Component
Type: IfcColumn
Tag: 47940
Global ID: 2U2Ls_mpn3vf7m$QruvLxU

Visual Quality Control properties

Predicted label:	Crack
Confidence level:	0.92
Material:	Concrete
Job_ID:	1
Capture_Device_ID:	DeviceName101
Processed_Data_Filename:	crack_1_processed.png

Visual Quality Control image

Figure 80 - Menu displaying a Visual QC result with detected defect ("Crack")



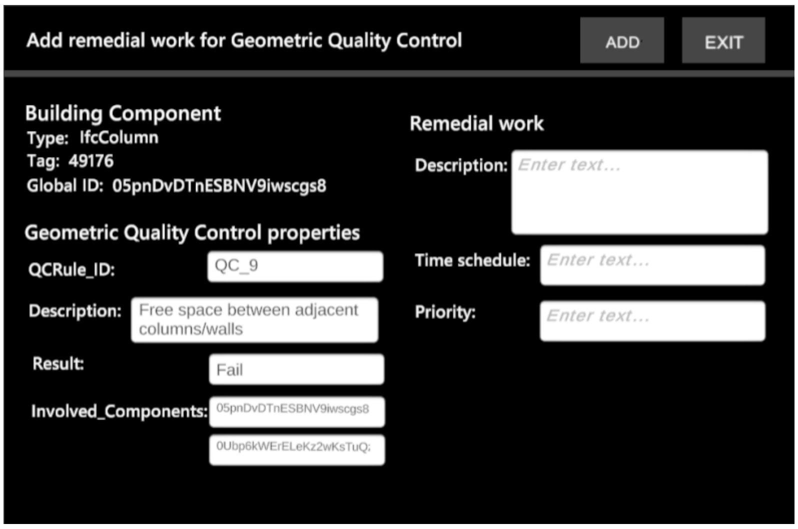
Building Component
Type: IfcColumn
Tag: 47940
Global ID: 2U2Ls_mpn3vf7m$QruvLxU

Visual Quality Control properties

Predicted label:	Crack
Confidence level:	0.92
Material:	Concrete
Job_ID:	1
Capture_Device_ID:	DeviceName101
Processed_Data_Filename:	crack_1_processed.png

Figure 81 - The user can switch between viewing the original and the annotated image from a Visual QC result

Add remedial work for QC result: After confirming the QC result by selecting the "Confirm" button in the *QC results notification* menu (in Figure 78 and Figure 80), the menu in Figure 82 opens-up so that the user can add a remedial work. The remedial work report contains three fields: 1) a description of the remedial work, 2) the time schedule, 3) the priority of the remedial work. The generated JSON file with the confirmed QC result and the generated remedial work is currently stored locally in the application. As the integration of the COGITO tools progresses, the file will be sent to the DT Platform using their provided API endpoints.



Building Component
Type: IfcColumn
Tag: 49176
Global ID: 05pnDvDTnESBNV9iwscgs8

Geometric Quality Control properties

QCRule_ID:	QC_9
Description:	Free space between adjacent columns/walls
Result:	Fail
Involved_Components:	05pnDvDTnESBNV9iwscgs8
	0Ubp6kWErELeKz2wKsTuQ:

Remedial work

Description:	Enter text...
Time schedule:	Enter text...
Priority:	Enter text...

Figure 82 -The user can add a remedial work to a confirmed QC result

Safety Control mode: If the user selects the Safety Control mode in the *Select mode* menu in Figure 62, the DigiTAR tool downloads the H&S results for the specific project from the DT Platform. In the second version of the DigiTAR tool, the Safety Control mode of the DigiTAR tool relies on an IFC enriched with information regarding safety spaces provided by the SafeConAI. The BIM model, as visualised in Unity, is depicted in Figure 83. In Figure 84, Figure 85 and Figure 86, a movement space, a fall space and a fall hazard space are depicted, respectively. The combination of movement spaces and fall spaces introduces, at the intersection, the fall hazards spaces. In Figure 87, the safety guardrails that are included in the safety enriched IFC are displayed. The user of the DigiTAR tool is notified when entering a fall space, as can be seen in Figure 88. While using the DigiTAR tool in potentially hazardous environments, the safety considerations in Section 2.4.3 should be taken into account.

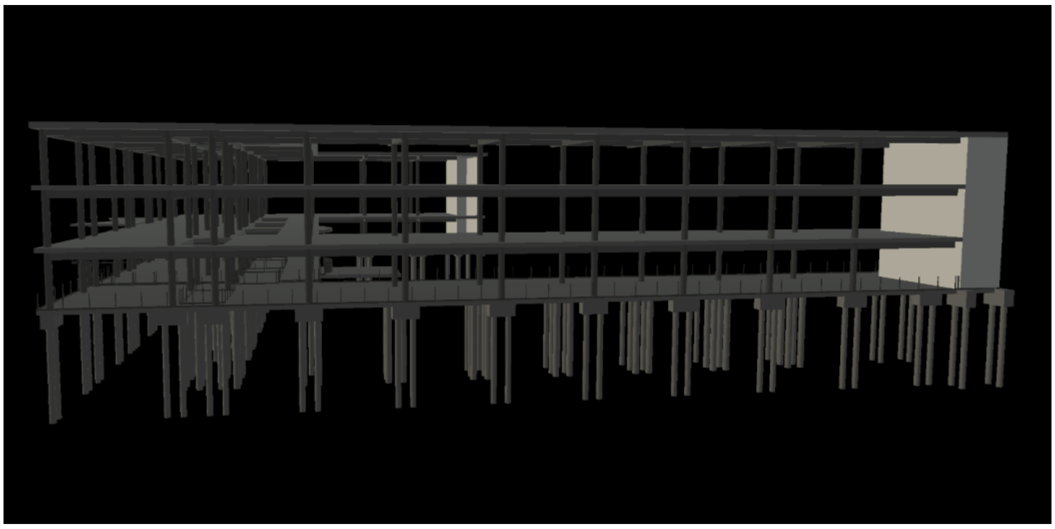


Figure 83 - The BIM model of the IFC file enriched with safety information regarding safe spaces as visualised in Unity

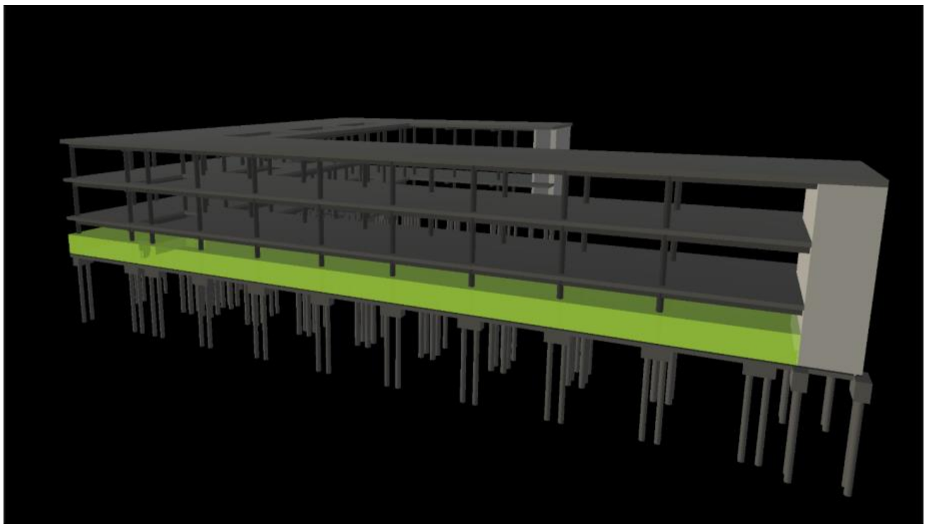


Figure 84 - Example of "Movement_space" as visualised in Unity

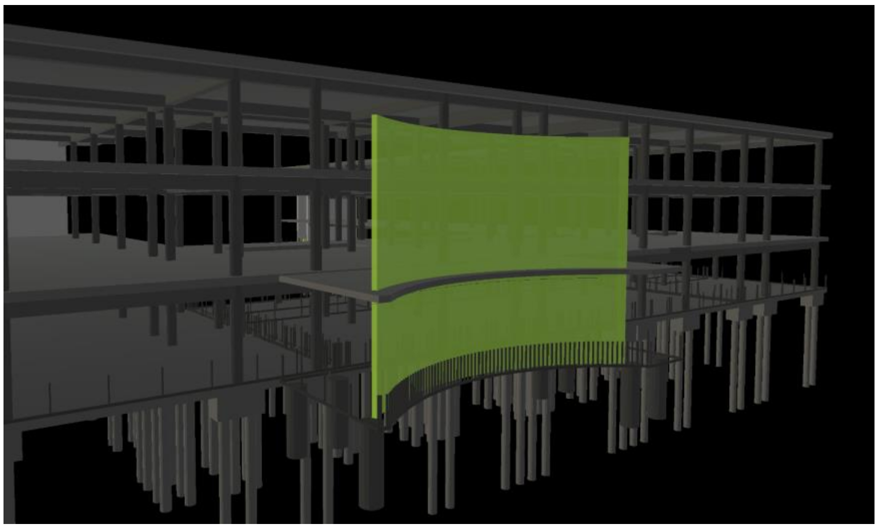


Figure 85 - Example of a "Fall" space as visualised in Unity

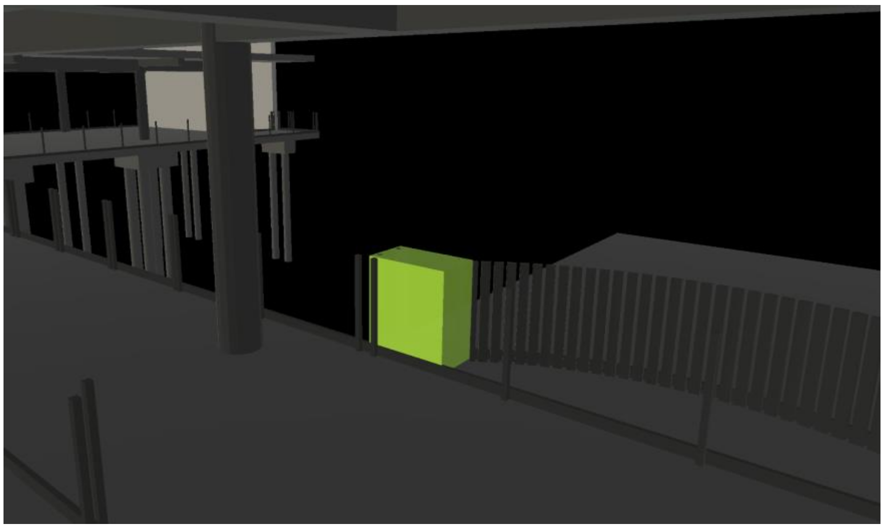


Figure 86 - Example of a “Fall_hazard" space as visualised in Unity

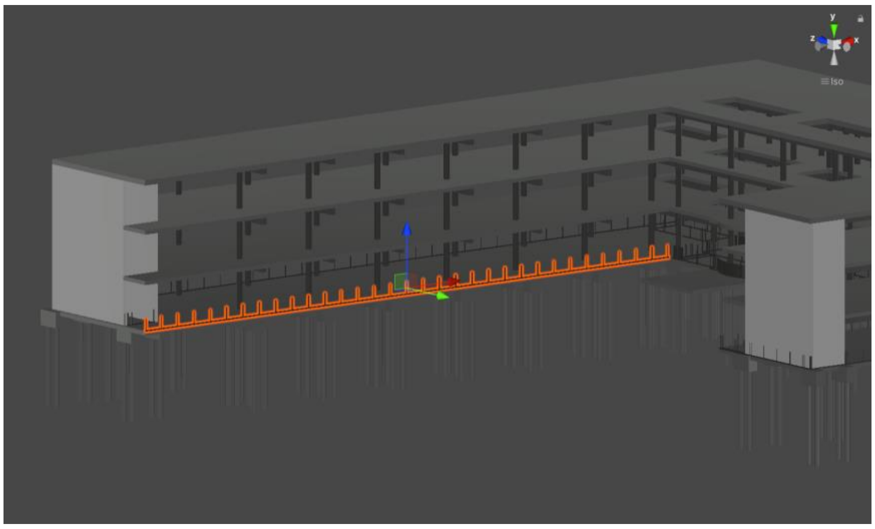
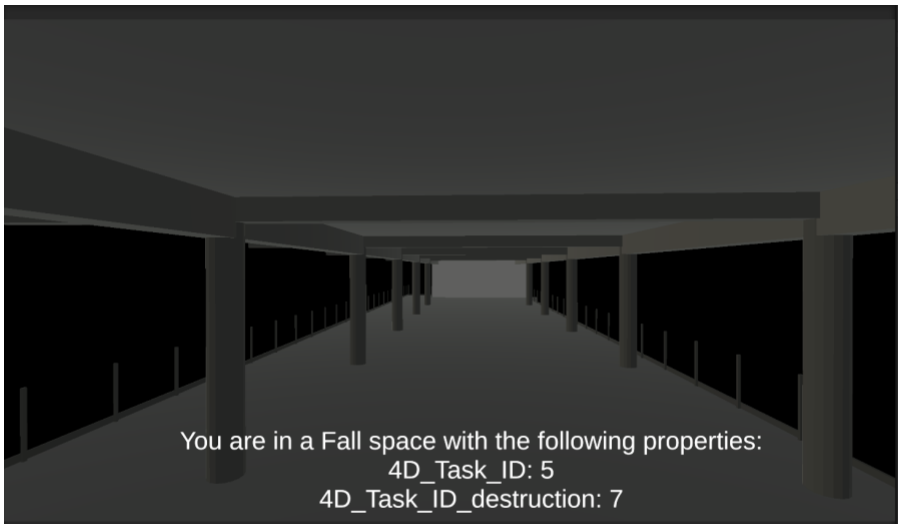


Figure 87 – Safety guardrails included in the safety enriched IFC, as visualised in Unity



You are in a Fall space with the following properties:
4D_Task_ID: 5
4D_Task_ID_destruction: 7

Figure 88 - The user is notified when entering a “Fall” space

2.4.3 Safety Considerations During Use

While wearing the HoloLens, the user of the DigiTAR tool walks around the construction site in order to examine the QC results, inspect the potential hazardous regions of the construction site and generate pre-processing jobs. Attention is required from the DigiTAR users, especially when using the application in spaces of the construction site that are not cleared from tripping hazards and/or do not have enough clear space for the user to move freely. Ideally, the DigiTAR tool should be used in environments with adequate light and plenty of space. Dark spaces may cause distraction and interfere with HoloLens calibration. The users of the DigiTAR tool (especially new users) are advised to take breaks periodically and rest if they experience any discomfort during the AR session. It may need some sessions for the user to get used to the AR experience. Correct calibration of the HoloLens device is recommended since it can decrease any discomfort created during an AR session.

2.5 Licensing

The DigiTAR module is a closed source component.

2.6 Installation Instructions

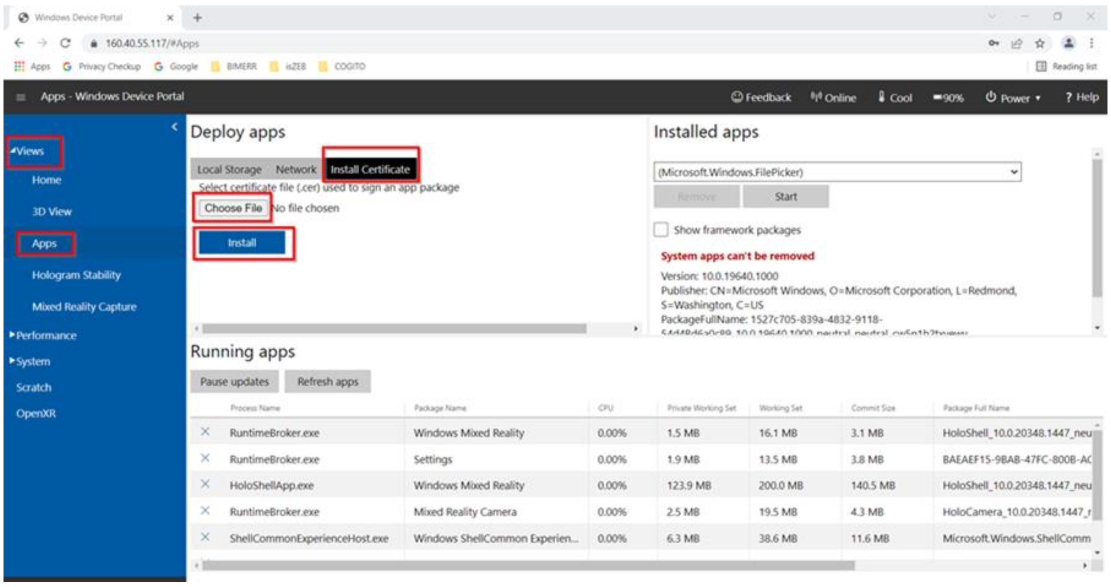
The DigiTAR tool can be installed via the Windows Device Portal of HoloLens. The detailed instructions for the user to enable the Windows Device Portal feature on HoloLens are[2]:

1. Power on the HoloLens and put on the device.
2. Use the Start gesture to launch the main menu.
3. Gaze at the “Settings” tile and select it using the Touch gesture or using a Hand ray.
4. Select the “Update & Security” menu item.
5. Select the “For developers” menu item.
6. Enable the “Use developer features”.
7. Scroll down and enable “Device Portal”.

Subsequently, to install the DigiTAR application:

1. First, the certificate file (.cer file) that accompanies the DigiTAR application should be installed. To install it, the user should open the Windows Device Portal and navigate to "Views", "Apps" and select "Install certificate". Subsequently, the user should select "Choose File" and browse to their computer to find the certificate file. The user should select it and then press "Install".

[2] https://docs.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/using-the-windows-device-portal#setting-up-hololens-to-use-windows-device-portal



Select certificate file (.cer) used to sign an app package

No file chosen

(Microsoft.Windows.FilePicker)

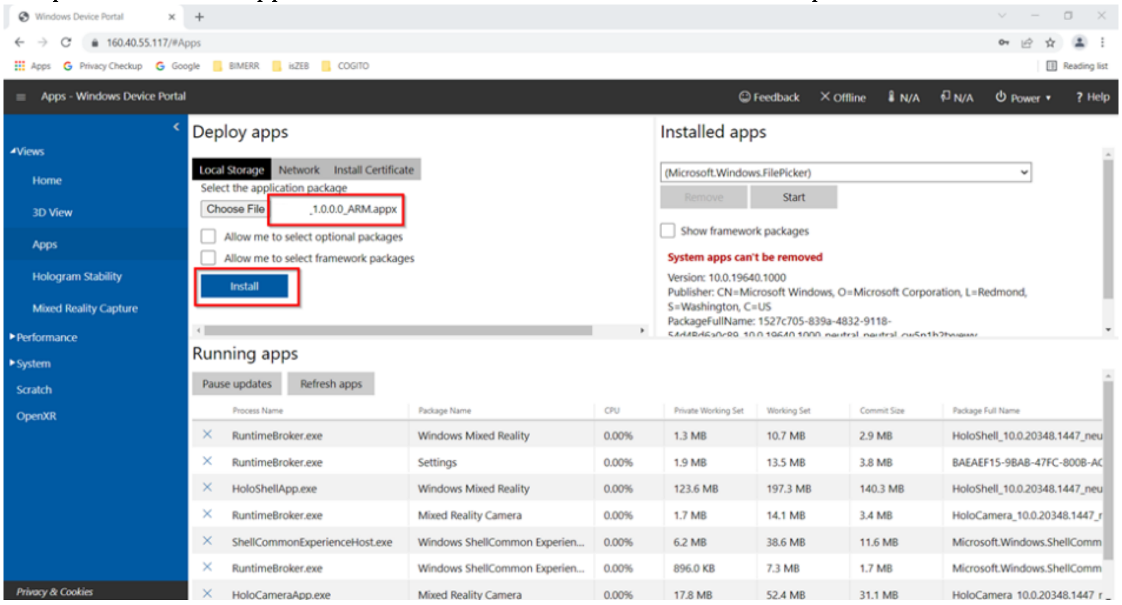
System apps can't be removed

Version: 10.0.19640.1000
Publisher: CN=Microsoft Windows, O=Microsoft Corporation, L=Redmond, S=Washington, C=US
PackageFullName: 1527c705-839a-4832-9118-

Process Name	Package Name	CPU	Private Working Set	Working Set	Commit Size	Package Full Name
RuntimeBroker.exe	Windows Mixed Reality	0.00%	1.5 MB	16.1 MB	3.1 MB	HoloShell_10.0.20348.1447_neu
RuntimeBroker.exe	Settings	0.00%	1.9 MB	13.5 MB	3.8 MB	BAEAEF15-9BAB-47FC-800B-AC
HoloShellApp.exe	Windows Mixed Reality	0.00%	123.9 MB	200.0 MB	140.5 MB	HoloShell_10.0.20348.1447_neu
RuntimeBroker.exe	Mixed Reality Camera	0.00%	2.5 MB	19.5 MB	4.3 MB	HoloCamera_10.0.20348.1447_r
ShellCommonExperienceHost.exe	Windows ShellCommon Experien...	0.00%	6.3 MB	38.6 MB	11.6 MB	Microsoft.Windows.ShellComm

Figure 89 - Certificate installation via Windows Device Portal

2. To install the application (.appx file), the user should open the Windows Device Portal and navigate to "Views", "Apps". From "Local Storage", they should select "Choose file" and browse to their computer to find the application file. The user should select it and then press "Install".



Select the application package

_1.0.0.0_ARM.appx

Allow me to select optional packages

Allow me to select framework packages

(Microsoft.Windows.FilePicker)

System apps can't be removed

Version: 10.0.19640.1000
Publisher: CN=Microsoft Windows, O=Microsoft Corporation, L=Redmond, S=Washington, C=US
PackageFullName: 1527c705-839a-4832-9118-

Process Name	Package Name	CPU	Private Working Set	Working Set	Commit Size	Package Full Name
RuntimeBroker.exe	Windows Mixed Reality	0.00%	1.3 MB	10.7 MB	2.9 MB	HoloShell_10.0.20348.1447_neu
RuntimeBroker.exe	Settings	0.00%	1.9 MB	13.5 MB	3.8 MB	BAEAEF15-9BAB-47FC-800B-AC
HoloShellApp.exe	Windows Mixed Reality	0.00%	123.6 MB	197.3 MB	140.3 MB	HoloShell_10.0.20348.1447_neu
RuntimeBroker.exe	Mixed Reality Camera	0.00%	1.7 MB	14.1 MB	3.4 MB	HoloCamera_10.0.20348.1447_r
ShellCommonExperienceHost.exe	Windows ShellCommon Experien...	0.00%	6.2 MB	38.6 MB	11.6 MB	Microsoft.Windows.ShellComm
RuntimeBroker.exe	Windows ShellCommon Experien...	0.00%	896.0 KB	7.3 MB	1.7 MB	Microsoft.Windows.ShellComm
HoloCameraApp.exe	Mixed Reality Camera	0.00%	17.8 MB	52.4 MB	31.1 MB	HoloCamera_10.0.20348.1447_r

Figure 90 - Application installation via the Windows Device Portal on HoloLens

3. The user is informed for the uploading and installation progress (see Figure 91), as well as when the installation is complete (see Figure 92).

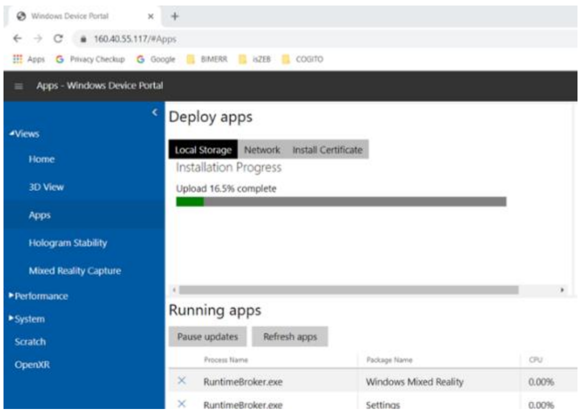


Figure 91 - Notification for installation progress

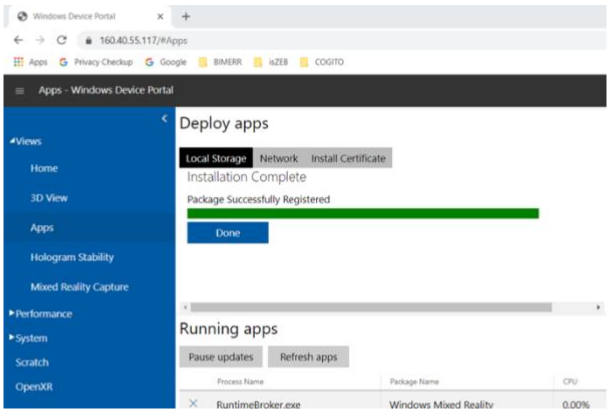


Figure 92 - Notification that installation is complete

2.7 Development and Integration Status

The DigiTAR module in this final release is considered developed, given that all the necessary functionalities have been implemented. The main functionalities for the Pre-processing mode, such as job addition, attachment of captured/selected images to a job, attachment of an IFC element to a job, filter selection, sending/receiving metadata and images with the Visual Data Pre-processing module, have been implemented and are ready to be used. The main functionalities for the Quality Control mode, such as on-site notification for the QC results, QC result confirmation and remedial work addition, have also been developed. The main functionalities for the Safety mode, such as the visualisation and parsing of the enriched with safety properties IFC has been developed in this version of the tool. In this final release of the DigiTAR tool, internal communication between the DigiTAR and the Visual Data Pre-processing module has been updated and refined. In this final release of the tool, the user authentication and authorization has been completely integrated with Keycloak. Preliminary communication has been established with the DT Platform for specific requests, such as for retrieving the list of projects, for downloading the BIM model, and for retrieving the QC results. During the integration phase, the communication and the data exchange between the DigiTAR and the DT Platform will be further tested, extended and optimized.

2.8 Requirements Coverage

Table 12 presents the stakeholder requirements as documented in D2.1 and are relevant to DigiTAR [11]. COGI-CS-2, COGI-CS-4 and COGI-CS-10 are merely covered by the programming language selected for the development of the DigiTAR tool. With regards to COGI-WF-2 and COGI-WF-4, the tool allows the Project Manager (PM) and Site Manager (SM)/Quality Manager (QM) to share information (design data, photos, point cloud etc.). In addition, concerning COGI-WF-5, the SM is able to share information with Subcontractors, Foreman and Workers (design data, photos etc.). Therefore, COGI-WF-2, COGI-WF-4 and COGI-WF-5 are going to be considered totally achieved at the end-to-end ICT COGITO System Integration.

Regarding the QC mode, COGI-QC-23, COGI-QC-24, and COGI-QC-25 are considered achieved since these functionalities have been developed in the DigiTAR tool. Finally, as concerns the H&S issues, COGI-SF-3 and COGI-SF-4 are partially achieved. Refinements and updates will be implemented during the integration process so that the user of DigiTAR can provide feedback on the hazard identifications and suggest mitigation works.

Table 12 - DigiTAR Stakeholders' requirements coverage from D2.1

ID	Description	Type	Priority	Status
COGI-CS-2	Runs on laptop and be usable on the construction site (remote access)	• Operational	Should	Achieved
COGI-CS-4	Runs on Windows	• Operational	Must	Achieved
COGI-CS-10	Allows visual comparison of current to planned status, either by AR glasses, mobile phones, or tablets	• Functional • Design constraint	Could	Achieved
COGI-WF-2	Allows the PM and Site Manager to share information (design data, photos etc.)	• Functional • Design constraint	Must	Partially achieved
COGI-WF-4	Allows the PM and Quality Manager to share information (design data, photos, design issues, schedules, work orders, materials schedule and usage, costs)	• Functional • Design constraint	Should	Partially achieved
COGI-WF-5	Allows the SM to share information with Subcontractors, Foreman, and Workers (design data, photos etc.)	• Functional • Design constraint	Should	Partially achieved
COGI-QC-23	Allows the QM to visualise and validate automated defect detections	• Functional • Design constraint • Performance	Must	Achieved
COGI-QC-24	Shows visual QC results using: graphic indicator; colourisation; and text	• Functional • Design constraint	Must	Achieved
COGI-QC-25	Shows visual QC defect with contextual information (link to components, defect history, etc)	• Functional • Design constraint	Should	Achieved
COGI-SF-3	Allows the HSE personnel to validate the automated hazard identification	• Functional • Design constraint	Could	Partially achieved
COGI-SF-4	Allows to employees to provide feedback and identify hazards	• Functional • Design constraint • Process	Should	Partially achieved

The functional and non-functional requirements were updated based on D2.5 “COGITO system architecture v2” [12] and are presented in Table 13. Concerning the functional requirements of the tool, Req.1-1 is covered while Req.1-2 is achieved, since the DigiTAR tool tracks every building component and defect that has been registered, as well as the safe/unsafe spaces. As for the Req.1-3, it is achieved, since currently the local position of the user in the 3D BIM model is determined. It should be noted that the final form of the coordinates may need to be adapted for the IFC files provided with georeferenced coordinates. Req-1.4 is achieved for the current form of received quality control defects but is partially achieved regarding the safety hazards. Req.1-5 and Req.1-6 are partially achieved since the generation of remedial works within the DigiTAR will be finalised during the integration phase. Req.1-7 is also partially achieved, since although the communication with the DT Platform is currently implemented for some interactions, it will be extended to cover all interactions in the T8.1 “End-to-end ICT System Integration, Testing and Refinement” (M18-M34). Finally, Req.1-8 and Req.1-9, that involve the communication between the DigiTAR and the Visual Data Pre-processing module, are achieved in this version of the tool.

Concerning the non-functional requirements, the status of Req-2.1 is considered to be achieved in this final version of the tool. Req-2.2 is partially achieved since it regards the WiFi connectivity while using the application on-site. Req-2.3, and Req-2.4 are covered since the DigiTAR is an AR application that already provides menus to let the user choose among the application's modes and functionalities. Req-2.5 is considered to be partially achieved, since although the architecture and technologies used in DigiTAR tool allow for both vertical and horizontal scaling [17], processing and resource demands may be restricted by the headset's computing capabilities. Req-2.6 can be considered as well covered by this version of the component, while Req-2.7 is achieved in this version of the DigiTAR tool since the integration with Keycloak has been established.

Table 13 – DigiTAR Functional and Non-Functional Requirements coverage from D2.5

ID	Description	Type	Status
Req-1.1	Maps the BIM 3D model enriched with defects and safety information, to the site.	Functional	Achieved
Req-1.2	Tracks every building component, defect and safety hazard that has been registered.	Functional	Achieved
Req-1.3	Determines user's position and orientation.	Functional	Achieved
Req-1.4	Confirms annotations about defects and safety hazards.	Functional	Partially achieved
Req-1.5	Creates annotations about safety hazards.	Functional	Partially achieved
Req-1.6	Creates task annotations for remedial work and safety hazard mitigation work.	Functional	Partially achieved
Req-1.7	Sends relevant information about annotations to DT platform.	Functional	Partially achieved
Req-1.8	Captures and sends visual data to Visual Data Pre-processing module.	Functional	Achieved
Req-1.9	Receives and displays processed visual data from Visual Data Pre-processing module.	Functional	Achieved
Req-2.1	User friendly interface.	Non-Functional	Achieved
Req-2.2	WiFi connection on-site.	Non-Functional	Partially achieved
Req-2.3	Menu to discern the different functionalities.	Non-Functional	Achieved
Req-2.4	AR Application.	Non-Functional	Achieved
Req-2.5	Scalability.	Non-Functional	Partially achieved
Req-2.6	Reusability.	Non-Functional	Achieved
Req-2.7	Security.	Non-Functional	Achieved

2.9 Assumptions and Restrictions

The final version of the DigiTAR tool is accompanied by certain assumptions and restrictions, which are presented in the following:

- The safety-related functionalities of the tool are based on the latest safety enriched IFC file, which is created by the SafeConAI and described in D4.2 "Preventive Health & Safety Application v2". The DigiTAR tool will adapt to any updates and modifications to these files during the System integration process.
- The final form of the coordinates to be sent with the generated within DigiTAR pre-processing jobs may need to be adapted, during the integration, for the IFC files provided with georeferenced coordinates.

- The communication between the DigiTAR tool and the DT Platform has already been established in this version of the tool for some interactions. However, the communication between the DigiTAR (as well as the rest COGITO components) and the DT Platform will be further extended, tested, and finalised, during the integration process (T8.1 End-to-end ICT System Integration, Testing and Refinement) in M18-M34.

3 Conclusions

This deliverable introduced the main functional components of the DigiTAR module and their use. Information regarding the development tools, the data exchange, and the API that the module uses were also presented.

The DigiTAR module is an AR application, which offers many functionalities to the user. The DigiTAR module has three different modes: the Visual Data Pre-processing mode, the Quality Control mode, and the Safety mode. Within the DigiTAR tool, the user is able to view the 3D BIM model on-site with the 3D IFC components overlaying the physical components. Due to the Data Collector functionality of the DigiTAR tool, the user can also capture image data on-site. The user can create pre-processing jobs, attach IFC elements to them, capture or select images to be linked to the pre-processing job and send job-related data and metadata to the Visual Data Pre-processing module. The user can view the processed/enhanced images within the DigiTAR tool and push them to the Visual Data Pre-processing module which further promotes them to the DT Platform, where they can be assessed by other COGITO components (Geometric and Visual Quality Control tools). The tool also notifies the user for Quality Control results and requests for on-site confirmation, along with the suggestion of remedial work. Finally, the DigiTAR module notifies the user for potential construction site hazards. In the current version of the tool, notification concerns fall spaces, but will be further extended based on the modifications made to the enriched with safety results IFC generated by the SafeConAI module.

The work presented here introduces a second version of the DigiTAR module. The module has been tested with sample data, e.g., sample QC results, sample IFC enriched with safety results. During the integration phase and as the COGITO tools evolve, additional updates and refinements will be performed, to support all the emerged demands as well as the connectivity and communication with the other COGITO tools. The DigiTAR module will be further tested on the pre-validation (T8.2) and validation (T8.4) sites with real case data during M21-30 and M28-34, respectively.

References

[1] S. Lockley, C. Benghi and M. Cerny, “Xbim.Essentials: a library for interoperable building information applications,” *Journal of Open Source Software,* vol. 2, no. 20, p. 473, 2017.

[2] [Online]. Available: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/linq/.

[3] [Online]. Available: https://standards.buildingsmart.org/.

[4] [Online]. Available: https://docs.unity3d.com/560/Documentation/Manual/Shaders.html.

[5] [Online]. Available: https://unity.com/.

[6] [Online]. Available: https://www.microsoft.com/en-us/hololens/hardware.

[7] [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/unity/mrtk-getting-started.

[8] [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/platform-capabilities-and-apis/using-the-hololens-emulator.

[9] [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/design/motion-controllers.

[10] [Online]. Available: https://docs.opencv.org/3.4/d3/dc1/tutorial_basic_linear_transform.html.

[11] D2.1-COGITO, “Deliverable 2.1: Stakeholder requirements for the COGITO system,” 2021.

[12] D2.5-COGITO, “Deliverable 2.5: COGITO System Architecture v2,” 2022.

[13] [Online]. Available: https://bimerr.eu/bimerr-tools/process-workflow-modelling-and-automation-toolkit-pwma/.

[14] [Online]. Available: https://microsoft.github.io/MixedRealityToolkit-Unity/Documentation/Updating.html#upgrading-to-a-new-version-of-mrtk.

COGITO

CONSTRUCTION PHASE
DIGITAL TWIN MODEL

cogito-project.eu

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 958310