V4Design

Visual and textual content re-purposing FOR(4) architecture, Design and virtual reality games

H2020-779962

D4.2

Basic version of interior and exterior localization algorithms &tools

Dissemination level:	Public
Contractual date of delivery:	Month 18, 30 June 2019
Actual date of delivery:	Month 18, 28 June 2019
Workpackage:	WP4 - 3D model extraction from 2D visual content
Task:	T4.2 Localization of the interior and exterior of buildings in visual content
Type:	Report
Approval Status:	Approved
Version:	1.0
Number of pages:	38
Filename:	D4.2-V4Design_Basic version of interior and exterior localization algorithms & tools_v1.0.pdf

Abstract

Basic version of interior and exterior localization algorithms & tools tested against the SoA on benchmark data with the first iterations of: (a) keyframe extraction, (b) shot detection, (c) object detection, and (d) scene recognition for building and interior object localization.

co-funded by the European Union

History

Version	Date	Reason	Revised by
0.1	06-04-2019	Table of content created & content defined	CERTH
0.2	17-06-2019	Finalized and released 1st version of the deliverable	CERTH
0.3	19-06-2019	New content for keyframe extraction and shot detection was included in the deliverable	KUL
0.4	22-06-2019	Release of the 2nd version of the deliverable, ready for internal review	CERTH
0.5	26-06-2019	Review 2nd version and recommend improvements	KUL
0.6	27-06-2019	Integrate recommendations and revise the document based on the improvements suggested in v0.5	CERTH
1.0	28-06-2019	Finalize and release final document to be submitted	CERTH

Author list

Organization	Name	Contact Information
CERTH	Elissavet Batziou	batziou.el@iti.gr
CERTH	Konstantinos Avgerinakis	koafgeri@iti.gr
CERTH	Stefanos Vrochidis	stefanos@iti.gr
KUL	Maarten Vergauwen	maarten.vergauwen@kuleuven.be
KUL	Jens Derdaele	jens.derdaele@kuleuven.be
KUL	Maarten Bassier	maarten.bassier@kuleuven.be

Executive Summary

This deliverable reports on the basic techniques for spatio-temporal building and object localization (STBOL). Specifically, this deliverable will elaborate on the initial methods for: (i) scene recognition, which also includes the methodology followed for keyframe extraction and shot-detection and (ii) spatio-temporal building and object localization in images and videos, which consists of two bilateral methods for object and building detection and localization. The goal of the spatio-temporal object and building localization component is to analyse the compiled visual data (images, videos) and provide on the one hand a computational boost to 3D reconstruction and on the other hand a semantic meaningful annotation for the observed visual objects and scenes.

The document elaborates on the WP4 modules, which are related to T4.2 and the appropriate approaches, components, and resources that were adopted in order to fulfil the respective functionalities that were described in the Description of Actions (DoA) and later on documented by the users throughout the compiled user requirements (D7.1, D7.2). The deliverable introduces the basic techniques for spatio-temporal building and object localization (STBOL) that were deployed during the first phase of the project's lifetime, for the implementation of the 1st prototype (M18). Furthermore, a description of the analysis requirements for visual components is provided and analysed thoroughly, while for each module an overview of the State-of-the-Art (SoA) and a comparison to other approaches is documented. The evaluation approaches and results are finally explained and demonstrated at the end of the document.

More specifically, the modules that are described in further detail are the ones that were deployed for fulfilling the basic functionalities of STBOL component:

a) The **Scene Recognition** (**SR**), which analyses visual data in order to segment the acquired video in shots and extract the most meaningful frames (i.e. keyframes) for 3D reconstruction and scene recognition. It also provides a high-level annotation about the existence of buildings and architectural objects that might exist in the visual scenes.

b) **Spatio-Temporal Object Localization** (**STOL**), which is responsible to detect, recognize and spatio-temporal localize the desired architectural objects that might exist in the acquired V4Design image and video samples.

c) **Spatio-Temporal Building Localization** (**STBL**) receives the scene recognition outcome, i.e. video frames and images that have a high probability to contain a building or architectural object and classifies them at a pixel-level in order to localize the architectural elements that exist inside them.

It is worth to note, that the performance of the above modules has been extensively evaluated in terms of their accuracy and the first experimental results are encouraging to continue to work on this direction.

Project partners, CERTH and KUL collaborated and deployed all the methodologies and modules for multimedia analysis and visual data understanding.

Abbreviations and Acronyms

ASPP	Atrous Spatial Pyramid Pooling
CH	Could Have
CNN	Convolutional Neural Network
ConvNet	Convolutional Neural Network
DoA	Description of Actions
CRF	Conditional Random Field
DCNN	Deep Convolutional Neural Network
DW	Deutsche Welle
GPU	Graphics Processing Unit
FAST	Features from Accelerated Segment Test
FC	Fully Connected
FCN	Fully Connected Network
FFmpeg	Fast Forward Moving Pictures Expert Group
FPN	Feature Pyramid Network
FR	Functional
HLUR	High Level User Requirement
JE	Joint Entropy
MH	Must Have
MI	Mutual Information
mIoU	Mean Intersection of Union
MS COCO	Microsoft Common Objects in Context
N-FR	Non Functional
RCNN	Region Convolutional Neural Network
ResNet	Residential Network
RoI	Region of Interest
SAD	Sum of Absolute Difference
SH	Should Have
SIFT	Scale Invariant Feature Transformation
SoA	State of the Art
SR	Scene Recognition
STBL	Spatio-temporal building localization
STBOL	Spatio-temporal building and object localization
STOL	Spatio-temporal object localization
SfM	Structure-from-motion
UR	User Requirement
VGG	Visual Geometry Group

Table of Contents

1 INTRODUCTION

In V4Design, the scope of T4.2 (Localization of the interior and exterior of buildings in visual content) is: (i) to identify buildings or objects of architectural interest in video frames and images in order to help the 3D reconstruction process (T4.3) to improve its computational efficiency and (ii) to segment these assets from their background content (image) so as to help 3D reconstruction (T4.3) to isolate the inlier from the outlier depth point clouds and recognize the segments within a set of predefined building classes in order to augment the BIM model (T4.4) that will be encapsulated within the Knowledge Base (T5.2). In general, the modules of T4.2 are essential and useful tools that help on the acceleration and semantic augmentation of the V4Design 3D models.

During the first half of V4Design project lifetime (M1-M18), T4.2 contributed on the first milestone (MS1) by defining the technical requirements of its modules (D6.1) and aligning them towards the initial user requirements (D7.1) that existed at that moment of the project. T4.2 also contributed to the second milestone (MS2) of the V4Design project by deploying and integrating the initial versions of scene recognition (SR) and spatio-temporal building and object localization (STBOL) to the operational prototype of the system (D6.3). At the same moment, T4.3 contributed to the definition of the updated technical requirements and system architecture (D6.2). Finally, T4.2 contributed to the third milestone (MS3) by improving spatio-temporal building localization (STBL) and continuing the integration of its modules for the implementation of V4Design first prototype (D6.4). The iteration of T4.2's implementation will continue until the completion of the final prototype, by contributing to the fourth (MS4) and final prototype (MS5). The described timeline is depicted in **Figure 1**.

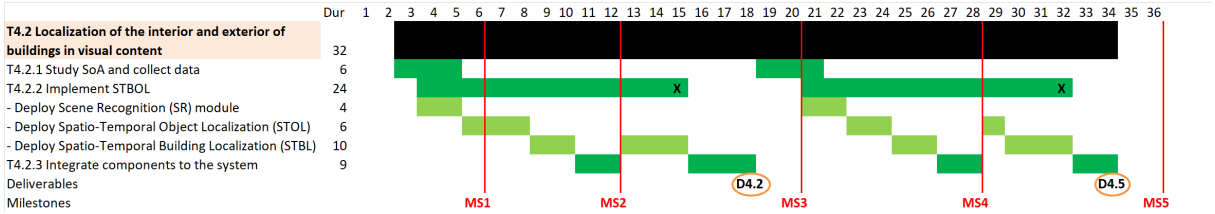


Figure 1: T4.2 towards V4Design lifetime

As described earlier, T4.2 interacts both internally and externally with Work-Packages (WP): WP2, WP4, WP5, WP6 and WP7. As far as internal interaction is concerned, T4.2 is responsible for the identification of the video frames or images that contain an exterior or interior architectural object, using Scene Recognition (SR), and for notifying T4.3, which is responsible for the reconstruction of 3D models from images or documentary videos, about their existence. This way, T4.3 can accelerate and improve the reconstruction procedure by ignoring the visual material that was deemed irrelevant. Further acceleration can be achieved by including Spatio-Temporal Building and Object Localization (STBOL), which extracts the background binary masks, and distinguishes between the foreground and background points that have been reconstructed. Finally, T4.2 interacts with T4.4 by providing the multi-class segmentation results that it acquires from STBOL and in this way can help the augmentation of the BIM models information.

As far as the external WPs are concerned, we identify the correlation between T4.2 and WP2 (T2.1, T2.3 and T2.4), which are responsible for crawling, scraping, identifying, annotating and accumulating all appropriate visual data that contain architectural structures and

interior artefacts. T4.2 not only uses these data to help and augment the 3D reconstruction process, but also to train and enhance its models. Additionally, T4.2 provides its scene recognition labels and localization/segmentation masks to T5.1 and T5.2 in order to populate the V4Design Knowledge Base (KB) and in this way augment the information that exists inside the KB and provide a more sophisticated retrieval process. T4.2 is finally related to T6.1, T7.1, T7.2, where the user and technical requirements are defined, and T6.4 where service integration is performed. Thus, it is clear that this task provides an essential and useful service for the V4Design platform and is intercorrelated with several tasks of the project.

1.1 Objectives

The objectives of T4.2 for the 1st period of the project (M1-M18) are aligned with the main goals that were described in the DoA and summarized as follows:

- Study the literature that exists on scene recognition and spatio-temporal building and object localization (Accomplished).
- Design and deploy the appropriate computer vision and deep learning algorithms that will determine the presence of buildings and interior objects in movies and documentaries to be re-used and re-purposed (Accomplished by implementing the SR module).
- Use and extend SoA technologies on keyframe extraction, shot detection, object detection and scene recognition in order to create a module for the spatio-temporal detection of interior objects and the classification of building exteriors, cityscapes and other built environments (Accomplished by the implementation of the STBOL module).

T4.2 also fulfilled several other goals in order to fulfil and satisfy all 1st year's reported use cases and user requirements (D7.1, D7.2):

- Not only distinguish whether buildings or architectural artefacts exist inside image and video frames, but also identify the scene where they exist. An initial estimation provides knowledge on whether the depicted scene is from an interior or exterior environment, while a more detailed estimation is given afterwards, predicting the class of the place.
- Distinguish between foreground and background pixels within a video frame or image, estimated that it contains a building, in order to identify and remove the outliers from the 3D point clouds and improve the computational cost of 3D reconstruction.
- Extract the segmentation mask of more than 100 building categories within an image and video frame, enhancing the BIM model by identifying the parts of the reconstructed buildings.
- Detect and provide semantic information about more than 500 interior objects that could be used as assets in the V4Design architecture and video game creation platforms.

1.2 Results towards the foreseen objectives of V4Design project

Until now, V4Design has fulfilled the foreseen objectives of the project by completing the development of the basic functionalities of scene recognition and spatio-temporal building and object localization with the following activities:

a) Gathered annotated visual data from benchmark datasets (EC002 – Wiki crawled images, EC006 – COCO dataset, EC007 – Open Images dataset, EC008 – ImageNet, EC009

– Places2 dataset, EC010 – SUN397, EC011 – Oxford buildings dataset, Mapillary road dataset) and V4Design consortium partners (IC001, IC002, IC005, IC006, IC007, IC008, IC009, IC010, IC011, IC012, IC013, IC014, IC015, IC016, IC017, IC019, IC020, IC021, IC022, IC023, IC024, IC025, IC026, IC028 as reported in D4.1) and used them so as to train their Scene Recognition (SR) and Spatio-Temporal Building and Object Localization (STBOL) modules.

b) Deployed the initial version of Scene Recognition (SR) in images and video frames from documentaries and drone footage by deploying State of the Art (SoA) computer vision and deep learning scene recognition algorithms in the compiled datasets.

c) Deployed the initial version of Spatio-temporal Building and Object Localization (STBOL) in images and video frames by deploying State-of-the-Art computer vision and deep learning object detection and scene segmentation algorithms.

1.3 Future plans

- To re-study the literature that exists on scene recognition and spatio-temporal building and object localization in order to update any advances and improvements that have been introduced in the literature.
- To gather novel visual annotated material and datasets to enhance the recognition and segmentation models that have been developed for SR and STBOL.
- To accelerate the computational efficiency of scene recognition module by redesigning and compressing its deep learning architecture based on the detected concepts.
- To extend spatio-temporal building and object localization by tracking and maintaining the coherency of detected objects and buildings throughout time.
- To design spatio-temporal masks that will maintain the temporal coherency of semantically segmented pixels over sequential video frames.

1.4 Outline

The outline of this deliverable is as follows. Sections 2 and 3 respectively contain a brief presentation of the relevant user requirements for the analysis of the visual and audio content for spatio-temporal building and object localization modules, and a description of the relevant state-of-the-art methodologies in the scientific fields of computer vision, deep learning and segmentation. The methodology analysis of the three modules, Scene Recognition (SR), Spatio-Temporal Object Localization (STOL) and Spatio-Temporal Building Localization (STBL) are then described in Sections 4, 5, 6 respectively. We decided to break Spatio-Temporal Building and Object Localization (STBOL) in two separate and distinguished algorithms, because they differ on the subject that they focus on (as highlighted in the related work), but also because their data require a totally different confrontation and analysis schema. Parameter selection, evaluation metrics and comparison to related work for all deployed algorithms is provided in Section 6, while Section 7 concludes the deliverable and defines the future work for SR and STBOL modules until M34.

2 SPATIO-TEMPORAL BUILDING AND OBJECT LOCALIZATION (STBOL) REQUIREMENTS

The V4Design user requirements have been identified in D7.2 Initial use case scenarios and user requirements. Some of them are associated with Scene Recognition (SR), while others are directly linked to Spatio-Temporal Building and Object Localization (STBOL). Further details of the former are provided in Section 2.1 , while the latter is broken down in Section 2.2 and Section, separating requirements for object localization from those for building localization. It will be clear from this section, that T4.2 is aligned with the user requirements that were defined in D7.2, as the basic version of its modules (SR, STBOL) already satisfies all of the defined direct and indirect needs.

2.1 Scene Recognition (SR) requirements

As can be seen in Table 1 four user requirements from D7.2 have been identified that can be directly or indirectly associated with the scene recognition module of V4Design, namely UR_10, UR_21, UR_50 and UR_66. As far as UR_10 and UR_21 are concerned, users required from the V4Design platform to extract meaningful tags and semantics data from the acquired video and image samples. Scene Recognition (SR) is a candidate module that can satisfy these criteria by providing meaningful building and architectural information about the data that V4Design's storage might contain. This mainly refers to recognition of interior/exterior and of the scene type in video and image samples but can also be correlated to 3D models through the SIMMO model and the Knowledge Base.

In UR_50 users defined that they would like to have access to lists of 3D models, but also find contextual information, other assets and related work through the V4Design platform. The SR module, through the visual analysis that it makes in videos and images, can provide this contextual information, by providing semantically useful information about which region of the picture or video frame belongs to the background and what other artefacts exist around the reconstructed 3D model through the tags that is feedforwards to the V4Design Knowledge Base (KB).

Finally, UR_66 defines that video game designers would like to have a list of references to existing 3D models that have been detected from a 2D video scene in order to directly import it into the authoring tool environment. Scene Recognition (SR) is not only able to identify the context of the scene, i.e. whether it is an interior or exterior space, but it can also classify the True Positive visual samples, i.e. the ones that are relevant to video game and architecture design, to a predefined list of desired places and produce tags for each one of them.

Table 1: Relevant user requirements reported in D7.2 for Scene Recognition (SR)

User Requirement (UR)	Associated HLUR	Detailed description	Functional or Non Functional (FR/N-FR)	Priority based on MoSCoW framework
UR_10	HLUR_203 HLUR_208	As a user I want further details about the acquired footage - image/ video	FR	MH

	HLUR_214	(semantic data/ tags)		
UR_21	HLUR_204	As a user I want a description of the acquired 3D model	FR	SH
UR_50	HLUR_201 HLUR_204 HLUR_208	As a user I would like to have access to lists of 3D models, but also find contextual information, other assets and related work	N-FR	SH
UR_66	HLUR_212 HLUR_215	As a game designer, I want to get a list of references to existing 3D models that have been detected from a 2D video scene to directly import into the authoring environment.	FR	SH

2.2 Spatio-Temporal Object Localization (STOL) requirements

As can be seen in Table 2, six user requirements from D7.2 have been associated with the Spatio-Temporal Object Localization (STOL) module of V4Design, namely the UR_10, UR_13, UR_21, UR_50, UR_63 and UR_66. As far as UR_10 and UR_21 are concerned, we can safely say that that contextual information and description about the objects that exist in the visual data is required. The STOL module satisfies this requirement, as it is able to recognize objects from the acquired images or video frames that have been identified from SR. This module indicates whether they contain an architectural object or building image or video frame and provides a tag for each detected object. The information is provided back to the V4Design system by informing the Knowledge Base (KB) with the appropriate tags and semantic data.

In UR_13, it is declared that the users would like to have further details about the bounding box of the extracted 3D model. The STOL module satisfies this requirement by providing multiple spatio-temporal bounding boxes for the detected objects that might appear in the visual scene, i.e. image or video. This is then correlated to the 3D model by feedforwarding the information through the V4Design platform and associating the data using the SIMMO model.

Then UR_50 identifies that V4Design users would like to have access to lists of 3D models, but also find contextual information, other assets and related work. STOL module can provide this contextual information from 3D models through tags, bounding boxes and masks that are acquired from the image and video data that it analyses.

In addition to all the above, in UR_63 it is declared that a film production company would like to be able to put a location to the assets, such as putting the asset in the exact place as intended. STOL can in cooperation with 3D reconstruction satisfy this requirement by providing the relative location of the detected assets through the bounding boxes and let the 3D reconstruction place the object in the 3D environment.

Finally, UR_66 declares that video game designers would like to get a list of references to existing 3D models that have been detected from a 2D video scene to directly import them into the authoring environment (i.e. Unity, Rhino-3D). STOL can satisfy this UR by

segmenting video or image samples and producing tags for each of the detected object. The V4Design platform can then be used to pass the label/tag to the appropriate 3D model.

Table 2: Relevant user requirements reported in D7.2 for Spatio-Temporal Object Localization (STOL)

User Requirement (UR)	Associated HLUR	Detailed description	Functional or Non Functional (FR/N-FR)	Priority based on MoSCoW framework
UR_10	HLUR_203 HLUR_208 HLUR_214	As a user I want further details about the acquired footage - image/ video (semantic data/ tags)	FR	MH
UR_13	HLUR_203 HLUR_208 HLUR_214	As a user I want further details about the bounding box of the extracted 3D model (unit independent)	FR	MH
UR_21	HLUR_204	As a user I want a description of the acquired 3D model	FR	SH
UR_50	HLUR_201 HLUR_204 HLUR_208	As a user I would like to have access to lists of 3D models, but also find contextual information, other assets and related work	N-FR	SH
UR_63	HLUR_212 HLUR_213 HLUR_215	As a film production company I want to be able to put location of the assets, such as putting the asset in the exact place as intended. A 3D drag-and-drop would be required	FR	CH
UR_66	HLUR_212 HLUR_215	As a game designer, I want to get a list of references to existing 3D models that have been detected from a 2D video scene to directly import into the authoring environment.	FR	SH

2.3 Spatio-Temporal Building Localization (STBL) requirements

As can be seen in Table 3, four user requirements from D7.2 have been associated with the Spatio-Temporal Building Localization (STBL) module of V4Design, namely the UR_10, UR_21, UR_50 and UR_66. Taking a closer look, we can see that STBL satisfies UR_10 and UR_21, by classifying each pixel of an image or video frame and providing a tag for each detected architectural object, to the V4Design system. In this way, the users are provided with valuable tags and descriptions about the assets that the V4Design data storage contains.

In UR_50, user partners declared that they would like to have access to lists of 3D models, but also find contextual information, other assets and related work. The STBL module provides this contextual information from 3D models through tags and masks, extracted from the analysis that it makes on the visual data that are also used for 3D reconstruction.

Finally, in UR_66, video game designers using the V4Design system declared that they want to have a list of references to existing 3D models that have been detected from a 2D video scene to directly import into the Unity authoring environment. STBL satisfying this by segmenting videos and images and producing tags for each detected class.

Table 3: Relevant user requirements reported in D7.2 for Spatio-Temporal Building Localization (STBL)

User Requirement (UR)	**Associated HLUR**	**Detailed description**	**Functional or Non Functional (FR/N-FR)**	**Priority based on MoSCoW framework**
UR_10	HLUR_203 HLUR_208 HLUR_214	As a user I want further details about the acquired footage - image/ video (semantic data/ tags)	FR	MH
UR_21	HLUR_204	As a user I want a description of the acquired 3D model	FR	SH
UR_50	HLUR_201 HLUR_204 HLUR_208	As a user I would like to have access to lists of 3D models, but also find contextual information, other assets and related work	N-FR	SH
UR_66	HLUR_212 HLUR_215	As a game designer, I want to get a list of references to existing 3D models that have been detected from a 2D video scene to directly import into the authoring environment.	FR	SH

3 RELEVANT WORK

During the first months of the first period (M3-M5), a thorough study of the relevant computer vision and deep learning domain has taken place. The study was mainly concentrated on the algorithms that were foreseen to been implemented during M1-M18, meaning Scene Recognition (SR) and Spatio-Temporal Building and Object Localization (STBOL). This involved the study of Deep Learning and Computer Vision algorithms that focus on scene recognition, object detection, and semantic segmentation. Relevant benchmark and accumulated within V4Design datasets are also reported and thoroughly analysed, in means of quality and applicability for each scenario, in D4.1.

3.1 Scene recognition (SR)

A recent study of the object detection literature indicates that significantly high results have already been achieved by using deep-learned features from convolutional neural networks (ConvNets) on large-scale object recognition datasets (ImageNet (Krizhevsky, 2012), MS COCO (Tsung-Yi Lin, 2014), etc.). However, since most Convolutional Neural Networks (ConvNets) features are designed for object detection, they cannot be directly used for scene classification, as desired for V4Design purposes because; the recognition mainly focuses on detecting small-scale rigid daily objects. To that end, we turned our attention to other techniques, such as (Zhou, 2017), which introduced a new scene-centric dataset called Places (Zhou, 2017) with more than 7 million images of scenes and focus on representing images through a holistic approach, trying to recognize the scene/place instead of separate objects. This dataset has been very useful and suitable for scene recognition problems, especially when trained on VGG16, VGG19 (Krizhevsky, 2012) and ResNet DCNN (He K. Z., 2016) architectures.

Another scene recognition technique, introduced in (Gong, 2014), applies ConvNets within local multi-scale patches and integrates the patch-based ConvNets with global ConvNets, in order to capture both detailed information and holistic characteristics in scenes. Moreover, in (Gangopadhyay, 2016), a statistical aggregation solution is proposed, based on ConvNets for scene classification. Both the convolutional neural networks (ConvNets) on large datasets (to acquire spatial information) and the resulting ConvNets features were further analysed by statistical methods in the temporal domain to maintain spatio-temporal coherency throughout their representation.

The acclaimed C3D feature was proposed in (Tran, 2015), and describes how to transform 2D ConvNets to 3D ConvNets in order to exploit deep convolutional information in both spatial and temporal dimensions. However, C3D can only handle small video clips with a few frames and discards the long-term information in videos, due to the large computational cost.

(Huang, 2019) focus, firstly, on the short-term motion and spatial properties, and secondly, on the long-term motion information. In this way, the method combines long-term information with short-term deep information, in order to obtain a complementary representation and better understanding toward scene recognition. In Section 4.1.3 we describe in detail the methodology of the implemented V4Design approach which is based on VGG-16 architecture combined with a subset of Places dataset.

3.2 Spatio-Temporal Object Localization (STOL)

Deep learning techniques have enormous success solving image segmentation problems and in image classification tasks. Fully Convolutional Networks for Semantic Segmentation, presented by (Long, 2015), popularized the use of end-to-end convolutional networks and introduced skip connections from higher resolution feature maps. (Lin G. a., 2017) propose to use an encoder part of ResNet-101 (He K. Z., 2016) blocks and a decoder part of RefineNet (Lin G. a., 2017) blocks, which concatenate high-resolution features from encoder and low-resolution features from previous RefineNet blocks. Another encoder-decoder architecture was proposed by (Peng, 2017) which includes very large kernels convolutions, but these large kernels convolutions are computationally expensive and they are adopted because networks tend to gather information from a smaller region. In the Mask R-CNN method (He K. a., 2017) a two stages approach is presented, based on Faster R-CNN (Ren, 2015) for object detection and localization. In the first stage, the candidate object bounding boxes are extracted using Region Proposal Networks. In the second one, the method extracts a binary mask for each region, in parallel to the feature extraction from each candidate bounding box and performs classification and bounding box regression. All the above-mentioned techniques are applicable to interior design object detection and localization while we focus on the Mask R-CNN approach.

3.3 Spatio-Temporal Building Localization (STBL)

Image segmentation approaches that are mainly related to buildings images have also been developed in literature. Scale Invariant Feature Transform (SIFT) descriptors are involved in (Shalunts, 2011), where a method is proposed to classify facade windows by their architectural style. The methodology is based on local feature learning and SIFT descriptor extraction, clustering them to learn a visual vocabulary. Other works in semantic segmentation from buildings, outdoors spaces and larger architecture structures, involve 3D models and point clouds as input, beyond the traditional 2D space of images and videos. (Martinovic, 2015) introduce a new approach for semantic segmentation of facade modelling based entirely on 3D models. Firstly, they present an image-based 3D point cloud and afterwards they classify and split the facade. The authors propose a structure-modelling step through architectural principles, before projecting original images onto the final estimated 3D model. (Liu, p. 2017) propose a symmetric regularization on the 2D facade parsing problem. The authors train an entirely deep convolutional neural network to mark bounding boxes that are generated by object detection. They apply their proposed symmetric loss for segmentation results and then they refine the results using Region Proposal Network (RPN) bounding boxes. In (Peng, 2017) the authors present a local classifier, which is learned to select views for multi-view semantic labelling. They find the single image part, which supports best the semantic labelling of each face of a 3D mesh model. (Gadde, 2017) propose an auto-context based framework for facade segmentation. This method is a sequence of decision tree classifiers that are stack pixel classifiers using auto-context features and 3D point clouds and then learned using stacked generalization. The methodology developed in V4Design is based on the “DeepLab” system (Chen, 2017), which is trained on images of buildings with semantic annotation. The method is learning to detect and localize items of buildings by applying the so-called “Atrous convolution” with up-sampled filters for dense feature extraction.

4 SCENE RECOGNITION (SR) - V1

The Scene Recognition (SR) focuses on analysing visual data (i.e. images, video frames), before 3D reconstruction module in order to provide not only the existence of buildings or architectural objects in a visual scene, but also identify whether the analysed frame depict an outdoors or indoors scene. This information will undoubtedly accelerate the computational efficiency of 3D reconstruction as it will not be obliged to analyse video frames or images that do not contain a target object or building. Furthermore, it will be easy to provide useful metadata to the appropriate 3D reconstruction in order to run the process for an interior or exterior environment. Scene recognition also provides the type of scene that is depicted in the video frame or image that is analysed, by choosing amongst the 365 category scenes that have been integrated in SR model until now, leading to useful annotations about the things that exist in the analysed videos and images.

As far as the SR modules are concerned, we deployed several types of techniques in order to analyse and process the acquired video frames and images. Initially, we deployed shot detection in order to segment the videos in semantically relevant video sequences, by tackling hard cuts and fade cuts as well. Blurred images were also identified so as not to be processed and spoil the scene recognition and 3D reconstruction analysis. The acquired images and video frames are analysed by Scene Recognition (SR) module, in order to understand the content and context of the analysed visual scene. In the following sections, we elaborate on the methodology that was followed in order to implement the basic functionalities of the aforementioned components.

4.1 Methodology

4.1.1 Shot Detection

In many cases videos and movies, directed by professionals, contain shots of multiple scenes and as such are not directly useful for photogrammetry. It is therefore important to pre-process these videos. In a preliminary step, the various shots are first delineated and can then be further processed to extract, proper frames for reconstruction (see Section 4.2 of D4.3).

Deliverable D4.3 explains in details that we implement two types of shot detection. A first algorithm deals with the detection of **hard cuts**. These are very common scene cuts where the transition between shots is abrupt. A typical example is shown in Figure 2. It was found that a SAD-based algorithm is fast and efficient and achieves better results than a histogram and mutual information-based method.

A second type of cuts is so-called **fade-cuts**. These transitions are harder to detect because they happen gradually. It is no surprise that mutual information methods achieve better results for this type of shot cuts, because they search for frames in which (part of) the information can be ‘explained’ by the previous or next frame. Especially the joint entropy (JE) value is a good discriminator and can be used for the detection of fade-cuts. A good example is shown in Figure 3.

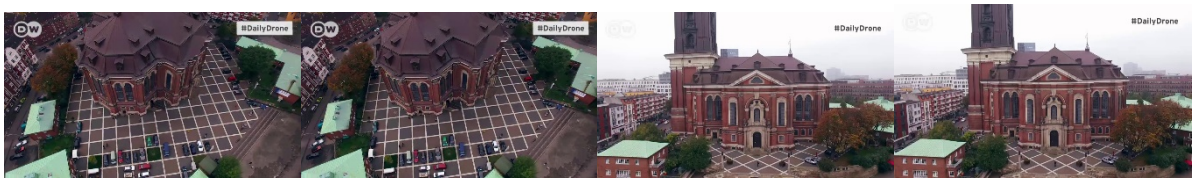


Figure 2: Consecutive frames 760, 761, 762 and 763 of the St Michael's church Daily Drone video. The scene cut between 761 and 762 is detected automatically.

Figure 3: Frames 2994 and 4765 where a fade cut is detected. These frames clearly contain information from two different shots, yielding large increases in the JE score.

4.1.2 Dealing with blurry frames

When video sequences are recorded in a freehand style, one can never totally exclude the possibility of blurred frames. In most cases this blurriness is actually motion blur that is the result of sudden motions of the camera. Blurred frames cause problems not only on scene recognition and spatio-temporal building and object techniques, but also on photogrammetric 3D reconstruction ones, mostly because the amount of matched or tracked features drops dramatically when such a frame is encountered. It is therefore best to try to detect these frames beforehand and deal with them.

Once more, details of the chosen algorithms are described in detail in Section 4.3 of deliverable D4.3. We found that two methods deliver useful results: a SAD-based algorithm that computes the total amount of gradient information in an image and compares it to neighbouring frames, and a keypoint-extraction based algorithm that compares the number of extracted features. The FAST feature detector was chosen because it is efficient, makes use of local intensity changes and does not suppress features in each other's neighbourhood. Figure 4 show examples of (parts of) detected blurry frames.

Figure 4: Two frames showing motion blur.

4.1.3 Scene recognition

As far as scene recognition is concerned, the methodology we adopted has Deep Convolutional Neural Networks (DCNNs) with two components: one on the hidden layers for the feature extraction part, and one for the classification part. In the feature extraction component, the network combines a sequence of convolution and pooling operations where the features are progressively detected. In the classification part, the fully connected layers serve as a classifier on top of these extracted features, assigning a probability for each class that the algorithm predicts.

Convolution is one of the main operations in a DCNN architecture, being the mathematical combination of two tensors to produce a third one. The convolution is performed on the input data with the use of a filter (known also as kernel) to then produce a feature map. We execute a convolution by sliding the filter over the input, which can be either a 2D or 3D array of elements. At every location, a matrix element-wise multiplication is performed and the result is summed onto the feature map. The output of the convolution is passed through an activation function. Stride is the step of the convolution filter displacement for each step and it is usually equal to one, meaning that the filter slides pixel by pixel.

In general, the size of the feature map is always smaller than the input; hence, it is common to prevent the feature map from shrinking using padding. After one or a stack of convolution layers, it is common to add one pooling layer to continuously reduce the dimensionality, thus reducing the number of parameters, to decrease the training time. The most frequent type of pooling is max pooling, which takes the maximum value in each considered window.

The convolution and pooling layers are then followed by a few fully connected layers (FC), which can only accept one-dimensional data. To convert our 3D feature array to one-dimensional vector we “flatten” the array by concatenating the rows of each dimension. This vector is further passed to a logistic regression classifier to produce the final vector of class score predictions. The input size of our training set is a set of m images with dimensions $n_h * n_w * \mathrm{n_c}$, where n_h and n_w are the height and the width of an image with n_c channels. VGG16 is a 16-layer neural network, not including the max-pool layers and the SoftMax activation in the last layer. In particular, the image is passed through a stack of convolutional layers, which are used with filters of a small receptive field $f * f$. Spatial pooling is carried

out by five max-pooling layers, which follow some of the convolutional layers (not all), as described in the original paper (Simonyan, 2014). Max-pooling is performed over a $w_p * w_p$ pixel window, with stride s.

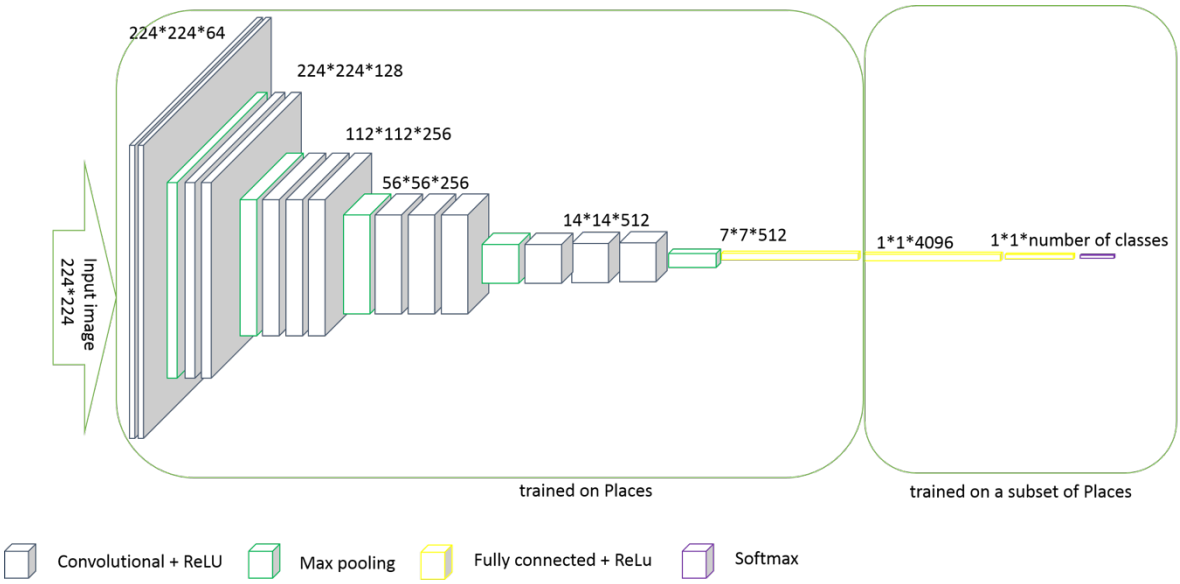


Figure 5: The framework of scene-recognition approach.

The width of convolutional layers starts from 64 in the first layers and then increases by a factor of 2 after each max-pooling layer, until it reaches 512 as depicted in Figure 5. The stack of convolutional layers is followed by three Fully-Connected (FC) layers. The final layer is the SoftMax layer. All hidden layers are equipped with the Rectified Linear unit (ReLU (Krizhevsky, 2012)), which is defined in Equation 1.

$$f(x) = -\max(0, x)$$

Equation 1: The Rectified Linear Unit

ReLU is an element-wise operation, applied per pixel, and replaces all negative pixel values in the feature map with zeros. The main property of ReLU is the introduction of non-linearity of the Convolutional Network and therefore the ability to identify and extract realistic non-linearity.

Innovation (beyond state of the art): In the context of V4Design Scene Recognition (SR) module, the VGG16 framework was pre-trained on Places dataset on first 14 layers, which has the initial 365 Places categories. The remaining layers were trained on a subset of 145 selected classes of Places dataset as depicted in the Table 4 in order to adjust the SR model to V4Design needs and classify only the relevant scenes. Moreover, Scene Recognition classifies scenes in two general environmental categories, i.e. indoor or outdoor, and use them so as to differentiate between frames that could create a 3D model for interior design (e.g. object) to a 3D model from exteriors (e.g. building). The great enhancement of this module is that in cooperation with 3D reconstruction and reasoning modules could lead to annotated 3D models that could be retrieved from the Knowledge Base (KB) and recreate fast and accurate interior and exterior 3D buildings and objects.

5 SPATIO-TEMPORAL OBJECT LOCALIZATION (STOL) - V1

Spatio-Temporal Object Localization (STOL) focuses on recognizing the architectural objects that exist in images and videos in a pixel level representation. These masks are then provided to 3D reconstruction model in order to be notified about the visual data that they contain potential object(s) that could be reconstructed and populated to the Knowledge Base (KB). In this way, 3D reconstruction module not only saves a great deal of processing time to analyse the number of video frames that exist in a video file, but also provides meaningful information to the Knowledge Base (KB) and the V4Design retrieval system. Authoring tools could also use these masks to highlight the objects that have been reconstructed and their predicted class name.

5.1 Methodology

The methodology that was followed so as to deploy STOL module of V4Design was initially based on (He K. a., 2017), which proposed a Mask RCNN approach to tackle pixel-wise object instance segmentation by extending Faster-RCNN (Ren, 2015). Mask RCNN adopts a two-stage pipeline, with the first stage identical to a Region Proposal Network (RPN). In the second stage, in parallel to predicting the class and box offset, Mask RCNN adds a branch, which outputs a binary segmentation mask for each Region of Interest (RoI). The new branch is a Fully Convolutional Network (FCN) on top of a CNN feature map. In order to avoid the misalignments caused by the original RoI pooling (RoIPool) layer, a "RoIAlign" layer is proposed to preserve the pixel-level spatial correspondence. With a backbone network ResNeXt101-FPN (Xie, 2017) (Lin T. Y., 2017), Mask RCNN achieves top results for the COCO object instance segmentation and bounding box object detection.

More specifically, the implementation of the initial version of the STOL module involves a training phase, which does not only take into consideration class label information and bounding box information about an object, but also mask information. The mask is the shape of the boundary of an object that includes additional information compared to the bounding box.

A multi-task loss on each sampled RoI is defined in Equation 2.

$$L = L_{cls} + L_{box} + L_{mask}$$

Equation 2: The definition of loss function.

The classification loss L_{cls} and bounding-box loss L_{box} are identical to those defined in (Girshick, 2015).

The mask part of the Equation 2 has a Km^2-dimensional output for each RoI, which encodes K binary masks of resolution $m \times m$, one for each of the K classes. L_{mask} is then defined as the average binary cross-entropy loss from a per-pixel sigmoid.

For a RoI associated with ground-truth class k, L_{mask} is only computed from the k-th mask and other mask outputs do not contribute to the loss.

A mask representation is used to encode an input object's spatial layout. Although class labels or box offsets collapse into short output vectors by fully connected (fc) layers, this is not the case with the spatial structure of the extracted masks. A $m \times m$ mask is predicted

from each RoI using an FCN (Long, 2015). In this way, each layer in the mask branch keeps the $m \times m$ object spatial layout without collapsing into a vector representation that lacks spatial dimensions. The network architecture, which we use, is the ResNet101, trained on the Microsoft Common Objects in Context (MS COCO) dataset. The overall framework is presented in Figure 6.

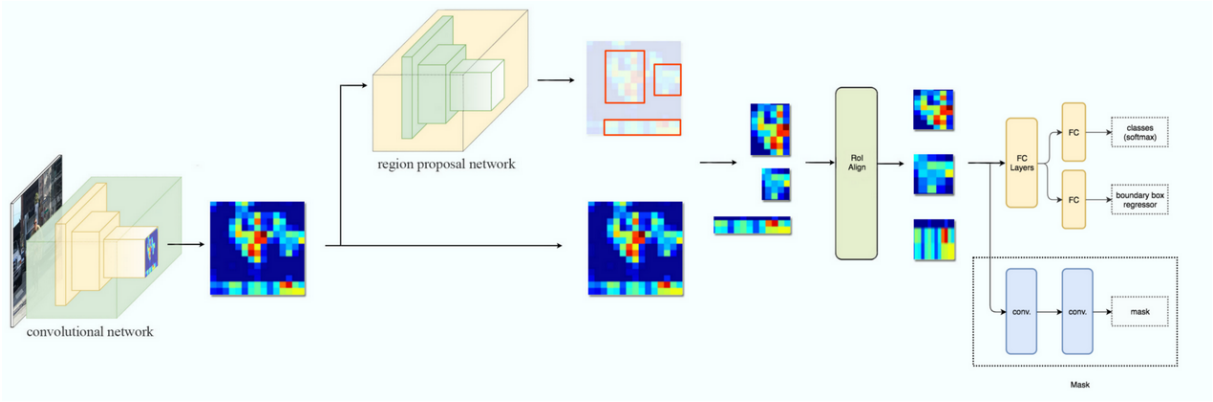


Figure 6: The framework of the developed Mask-RCNN method.

In addition, it is described the "RoIAlign" layer, which is developed because it is critical in mask prediction. RoIAlign is based on RoIPool (Girshick, 2015), which is a standard operation for extracting a small feature map (e.g., 7×7) from each RoI. RoIPool first quantizes a floating-number RoI to the discrete granularity of the feature map. The output, i.e. the quantized RoI, is then subdivided into spatial bins, which are themselves quantized, and finally feature values covered by each bin are aggregated by max pooling. A negative aspect of RoIPool is the quantization that is performed, e.g., on a continuous coordinate x by computing $[x/16]$, where 16 is a feature map stride and $[\cdot]$ is rounding. Similarly, quantization is performed within dividing into bins (e.g., 7×7).

Contrary to RoIPool, RoIAlign removes the harsh quantization of RoIPool, properly aligning the extracted features with the input. Bi-linear interpolation (Jaderberg, 2015) is used to compute the exact values of the input features at four regularly sampled locations in each RoI bin, and aggregate the result (using max or average). In this manner, quantization of the RoI boundaries or bins is avoided, using $x/16$ instead of rounding $[x/16]$.

Innovation (beyond state of the art): As far as spatio-temporal object localization (STOL) module is concerned, CERTH deployed a mask-RCNN framework, which adopts the two-stage pipeline. The initial stage is identical to a Region Proposal Network (RPN), while the second one, runs in parallel with the former so as to predict the class and box offset. Mask RCNN adds a branch that outputs a binary segmentation mask for each one of the detected Region of Interest (RoI). The new branch is a Fully Convolutional Network (FCN) on top of a CNN feature map. In order to avoid the misalignments caused by the original RoI pooling (RoIPool) layer, a "RoIAlign" layer is proposed in order to preserve the pixel-level spatial correspondence. The V4Design STOL module uses a backbone network ResNeXt101-FPN and fine-tuned Mask RCNN on the MS-COCO dataset in order to come in align with V4Design project and localise interior objects and provide segmentation masks. This not only improved the computational efficiency of STOL algorithm but also improved its prediction scores in the remaining objects.

6 SPATIO-TEMPORAL BUILDING LOCALIZATION (STBL) – V1

Spatio-Temporal Building Localization (STBL) uses the video frames outcome from scene recognition module, which probably have a high probability to contain a building or architectural object, in order to extract their segmentation mask by classifying them at a pixel-level. The goal of this component is to provide binary masks to 3D reconstructions so as to initially segment the foreground from the background elements of a visual scene and afterwards to annotate the 3D models with appropriate recognized architectural structure.

6.1 Methodology

The STBL module of V4Design is based on DeepLab (Chen, 2017) that allows for segmenting images and visual content in general. The model is adapted to the needs of V4Design by selecting a specific set of classes to learn. Atrous convolution (Equation 3) is also involved, that explicitly controls the solution at which feature responses are computed within deep convolutional neural networks. The method involves enlargement of the view of filters so as to incorporate larger context without increasing the number of parameters or computation time. The Atrous spatial pyramid pooling (ASPP) is used to segment objects, such as items of buildings, at multiple scales. Another strength of DeepLab is that it improves the localization of object boundaries by combining methods from DCNNs and a fully connected conditional random field (CRF), which is shown both qualitatively and quantitatively to improve localization performance. The framework of DeepLab model is illustrated in Figure 7 and, in the following, we describe the adopted methodology.

In the first step or the method, the input image goes through the network with the use of Atrous convolution and ASPP. The output from this network is a bilinear interpolation and, in the second step, it goes through the fully connected CRF to then fine-tune the result and get the final output. The equation of Atrous convolution is shown in Equation 3.

$$y[i] = \sum_{k=1}^{K} x[i + r * k]w[k]$$

Equation 3: The equation of Atrous convolution

When $r = 1$ (r = the rate), it is the standard convolution and when $r > 1$, it is the Atrous convolution, which is the stride to get the input sample during convolution. Atrous convolution allows enlarging the field of view of filters to incorporate larger context. It thus offers an efficient mechanism to control the field-of-view and finds the best trade-off between accurate localization (small field-of-view) and context assimilation (large field-of-view).

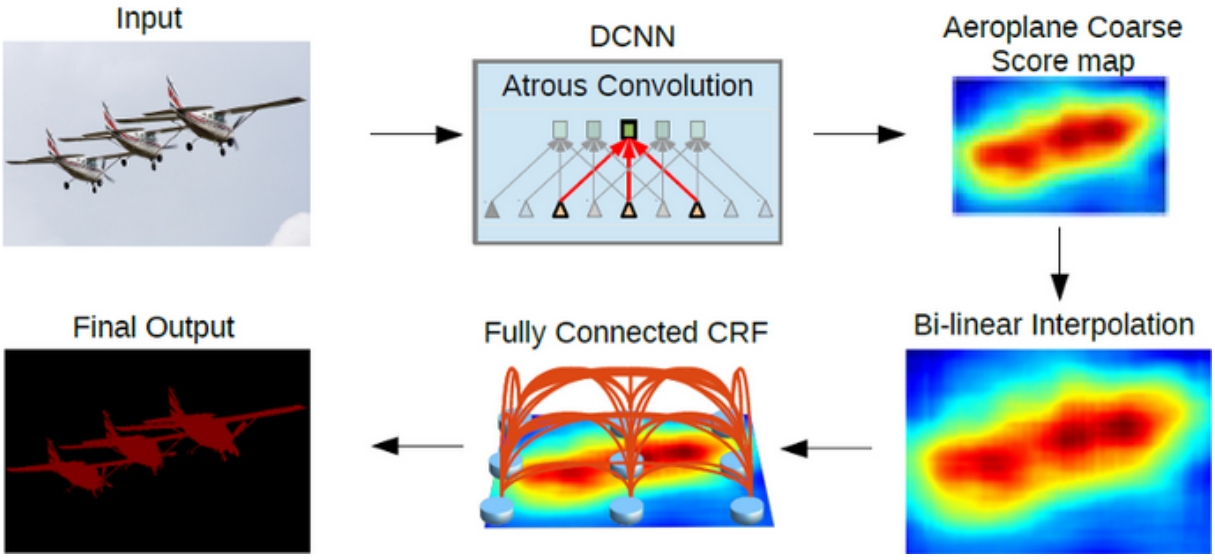


Figure 7: The framework of DeepLab method

ASPP actually is an Atrous version of SPP, for which the concept was introduced in (He K. Z., 2015). In ASPP, parallel Atrous convolutions with different rates are applied in the input feature map, and are fused together. As objects of the same class can have different scales in the image, ASPP helps to account for different object scales, which can improve the accuracy.

A Fully Connected Conditional Random Field (FC-CRF) is applied to the network output:

$$E(x) = \sum_i \theta_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j)$$

$$\theta_i(x_i) = -logP(x_i)$$

$$\theta_{ij}(x_i, x_j) = \mu(x_i, x_j)[w_1 \exp\left(\frac{\left||p_i - p_j|\right|^2}{2\sigma_\alpha^2} - \frac{\left||I_i - I_j|\right|^2}{2\sigma_\beta^2}\right) + w_2 \exp\left(-\frac{\left||p_i - p_j|\right|^2}{2\sigma_\gamma^2}\right)]$$

Equation 4: Fully connected Conditional Random Field.

Where x is the label assignment for pixels and $P\ (x_i)$ is the label assignment probability at pixel i. Therefore, the first term θ_i is the log probability. The second term, θ_{ij}, is a filter, in which $\mu = 1$ when $x_i \neq x_j$, and $\mu = 0$ when $x_i = x_j$.

In the brackets of the last equation in Equation 4, we see the weighted sum of two kernels. The first kernel depends on pixel value difference and pixel position difference, which is a kind of bilateral filter, having the property of preserving edges. The second kernel only depends on pixel position difference, which is a Gaussian filter. The optimal σ and w are found by cross validation. We choose to set the number of the iterations to 10, as we believe that this will give a quite good prediction of the two parameters.

In the context of V4Design's Spatio-Temporal Building Localization module, the aforementioned framework is trained on a subset of 17 classes of Mapillary dataset (Neuhold, 2017) and is learning to detect buildings, architectural structures, as well as to localize them.

Innovation (beyond state of the art): As far as Spatio-temporal building localization (STBL) module is concerned, Deeplab framework was adopted and trained with V4Design aligned and selected architectural elements from Mapillary dataset, in order to segment and recognise the potential objects in the provided visual content. The implemented Deeplab method uses Atrous Convolution, Fully Connected Conditional Random Field (CRF) and Atrous Spatial Pyramid Pooling (ASPP) for acquiring the appropriate segmentation masks. Experimental work took place with different DCNN architecture schemes, such as ResNet and VGGNet, leading to the best acquired implementation for STBL purposes. Differently to SoA, STBL concentrated only to architectural structures and elements, leading not only to computationally efficient results, but also to highly accurate predictions.

7 EVALUATION

7.1 Scene Recognition

7.1.1 Dataset

The Places dataset contains 1,803,460 training images with the image number per class varying from 3,068 to 5,000. Examples of different categories from Places dataset are illustrated in Figure 8. The validation set has 50 images per class and the test set has 900 images per class. Following the user needs of V4Design, we focus on specific classes as depicted in Table 4 so as to improve the performance of the model both in terms of efficiency and effectiveness.

Figure 8: Examples of different categories from Places dataset

Table 4: The 145 selected classes of Places dataset that were used in V4Design.

alcove	castle	excavation	library_indoor	pub_indoor
Alley	catacomb	fabric_store	library_outdoor	restaurant
amphitheater	cemetery	farm	lighthouse	restaurant_kitchen
amusement_park	chalet	fire_escape	living_room	restaurant_patio
apartment_building_outdoor	childs_room	fire_station	lobby	River
aqueduct	church_indoor	food_court	mansion	rock_arch
arcade	church_outdoor	formal_garden	manufactured_home	rope_bridge
Arch	classroom	garage_indoor	market_indoor	Ruin
archaelogical_excavation	clean_room	garage_outdoor	market_outdoor	Schoolhouse
archive	corridor	gazebo_exterior	mausoleum	Shed
atrium_public	cottage	general_store_indoor	mosque_outdoor	Shopfront
Attic	courthouse	general_store_outdoor	motel	shopping_mall_indoor
auditorium	courtyard	greenhouse_indoor	movie_theater_indoor	ski_resort
balcony_exterior	crosswalk	greenhouse_outdoor	museum_indoor	Skyscraper
balcony_interior	department_store	gymnasium_indoor	museum_outdoor	Stable
Bar	diner_outdoor	home_office	nursery	swimming_pool_indoor

barn	dining_hall	home_theater	oast_house	swimming_pool_outdoor
barndoor	dining_room	hospital	office_building	synagogue_outdoor
bathroom	discotheque	hospital_room	pagoda	temple_asia
bazaar_indoor	doorway_outdoor	hotel_outdoor	palace	throne_room
bazaar_outdoor	downtown	hotel_room	pantry	Tower
beach_house	driveway	house	park	train_station_platform
bedroom	elevator_door	hunting_lodge_outdoor	parking_garage_indoor	tree_house
berth	elevator_lobby	igloo	parking_garage_outdoor	Village
bow_window_indoor	elevator_shaft	industrial_area	parking_lot	water_tower
bridge	embassy	inn_outdoor	patio	wind_farm
building_facade	engine_room	kasbah	pavilion	Windmill
cafeteria	entrance_hall	kindergarden_classroom	pier	youth_hostel
campus	escalator_indoor	kitchen	playground	zen_garden

7.1.2 Comparison with existing tools in the market

There isn't a specific market that focuses on scene recognition but there is some research carried out and presented in State-of-the-Art papers for general scene recognition purposes. The tendency in this research domain is to use alterations of DCNN frameworks, such as AlexNet, GoogLeNet and ResNet (Zhou, 2017). Contrary to these approaches the V4Design Scene Recognition module using a VGG-16 architecture, which specialises and is trained only in classes that are of interest to architectural structures and elements. The comparison of V4Design SR module to these techniques follows in Section 7.1.3

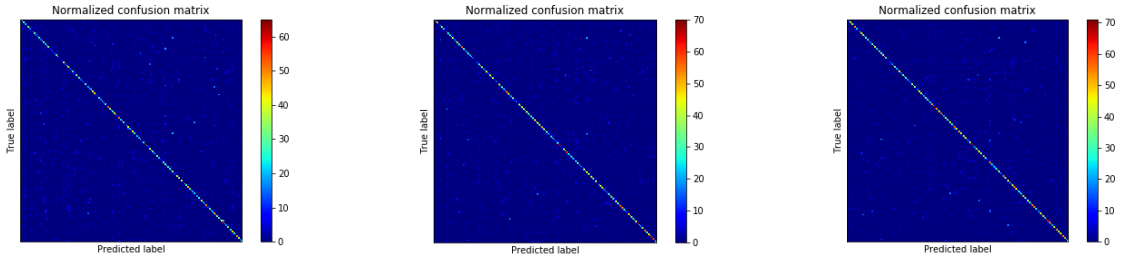
7.1.3 Evaluation

Based on the scene recognition methodology we have described in Section 4.1.3 , the model is pre-trained on the Places dataset for the first 14 layers and on a subset of the Places dataset for the last two layers. For the evaluation of the scene recognition module, we examined several combinations of parameter settings in order to select the best performing model. The selected parameters and their results are presented in Table 5. Moreover, in Figure 9 we present the corresponding confusion matrices.

Table 5: Results of the scene recognition experiments.

	Layers trained	Batch size	Epochs	Acc 1	Acc 5
Run1	4	32	20	37,42%	65,42%
Run2	2	32	20	44,55%	72,06%
Run3	**2**	**64**	**10**	**50,17%**	**78,31%**

Moreover, the integrated scene recognition module (run3) is not only annotated by the class label, but also discriminates this class in a broader category as indoor or outdoor.



Run 1 Run 2 Run 3

Figure 9: Confusion matrices of SR experiments.

The evaluation involves analysis on V4Design visual data, as given by the project's content providers. The results are demonstrated in Figure 10, Figure 11 and Figure 12.

In the categorized dataset classes as indoor or outdoor environment, we examine if the algorithm is able to predict this feature from the video content of DW's daily drone imagery. In Figure 10 the first two photos correctly predict the content as indoor and the rest as outdoor. The first photo (up, left) is classified as dining room, while the second photo (up, right) as attic. Despite the fact that the two video frames are semantically close to each other, the second frame has some features (e.g. the slope of the left wall) that direct the observer to an attic. The third frame is marked correctly as a balcony and the fourth one as village. The last set of semantically similar frames involves again two different predictions: as house on the left and a motel on the right. Even for a human observer is not possible to decide whether the photo depicts a house or a motel.

Figure 10: Frames from video of content provider.

In Figure 11 and Figure 12 we present qualitative results of the SR module from Gendarmenmarkt pictures and daily drone video frames. We observe that some images depict almost the same content with different output predictions such as village, palace and downtown in Figure 12 or palace, church and tower in Figure 11. However, making the distinction between these classes is a challenging problem even for a human.

Figure 11: Results of SR module for Gendarmenmarkt.

Figure 12: Results of SR module of Daily drone video frames.

7.2 Spatio-Temporal Object Localization (STOL)

We have described in Section 5.1 an implementation of Mask R-CNN, where the model generates bounding boxes and segmentation masks for each instance of an object in the input image. The model is based on Feature Pyramid Network (FPN) and a ResNet101 architecture.

7.2.1 Dataset

The COCO dataset (Tsung-Yi Lin, 2014) contains 91 common object categories (examples are illustrated in Figure 13), with most of them having more than 5000 labelled instances. In total, the dataset has 2,500,000 labelled instances in 328,000 images. In contrast to other benchmark datasets, COCO dataset has fewer categories but more instances per category and per image.

Figure 13: Images from different categories from MS COCO dataset.

7.2.2 Comparison with existing tools in the market

After a thorough investigation of the spatio-temporal object localization market, we found that there are some tools already exist there. More specifically, we can find on Google Play and AppleStore that there are some mobile applications that have already included deep learning models, pre-trained on benchmark object datasets. Object detector and iDetection are two examples of this kind of applications. Object detector uses 80 classes of COCO dataset and 1000 classes of Imagenet without bounding box. iDetection uses the YOLOv3-SPP object detection algorithm. However, the V4Design STOL module uses more advanced algorithms, such as Mask-RCNN variations trained in MS-COCO dataset, specifically tailored on V4Design purposes, revolving architecture and video game design. In addition, we provide a segmentation mask for each one of the detected objects. Further investingation and comparison with the baseline techniques of these tools follows in Section 7.2.3

7.2.3 Evaluation

The implementation of our V4Design approach follows the Mask RCNN method, as described in Section 5.1 , but there are a few cases where we improved or extended the implementation. Firstly, we resize all images to the same size so as to support training multiple images per batch. We preserve the aspect ratio, i.e. when the image is not square, we pad it with zeros. Secondly, we use smaller learning rates than the original implementation of 0.02, because we observed that smaller learning rates converge faster.

In Figure 14 and Figure 15 we present some qualitative results from images and videos that content providers give us in style object localization. In the case of Figure 14 the objects as table, chair, glass and persons are segmented correctly, but as can be seen in the Figure 15 there are some examples in which the module was not able to segment some objects in the desired way. In particular, in the first and third images of Figure 15, the module falsely predicts the statue and the pictures of people respectively as persons, which is not so absurd since both statue and picture depict human faces. In the case of the second image, a part of the carpet's pattern is segmented as a book, but we realize that this specific area has the shape and the colour that could potentially be a book.

Figure 14: Truly Segmented images using the V4Design STOL implemented approach

Figure 15: Falsely Segmented images using the V4Design STOL implemented approach.

7.3 Spatio-Temporal Building Localization (STBL)

7.3.1 Dataset

The Mapillary Vistas Dataset (Neuhold, 2017) is a large-scale street-level dataset containing 25,000 high-resolution images annotated into 66 object categories with additional, instances-specific labels as presented in Figure 16.

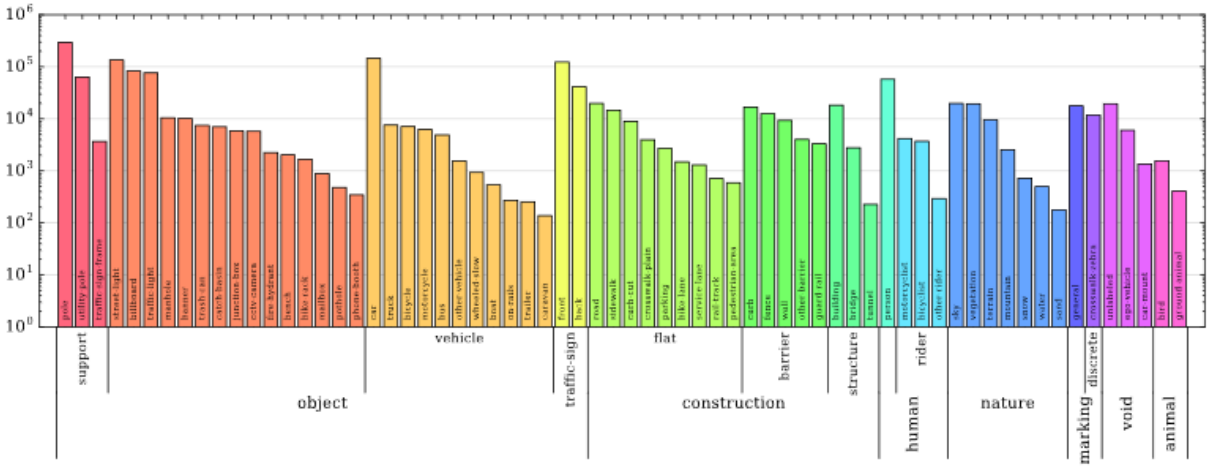


Figure 16: All categories of Mapillary Vistas dataset.

In our case, we omit some classes like nature or vehicle, since we focus on architectural structures. Our developed DeepLab model is fine-tuned on the Mapillary dataset, but only for the labels, which are included in the construction category, while all the others are concerned as background. This not only produces a computational efficient system that concentrates on the detection of architecture related artefacts, but also increases the classification accuracy of the model as it has fewer classes to distinguish from each other. The labels that we use in V4Design module are presented in the Table 6.

Table 6: The selected categories of Mapillary Vistas dataset

Class names	Class id
barrier--curb	1
barrier--fence	2
barrier--guard-rail	3
barrier--other-barrier	4
barrier--wall	5
flat--bike-lane	6
flat--crosswalk-plain	7
flat--curb-cut	8
flat--parking	9
flat--pedestrian-area	10
flat--rail-track	11
flat--road	12
flat--service-lane	13
flat--sidewalk	14
structure--bridge	15
structure--building	16
structure--tunnel	17
other	0

7.3.2 Comparison with existing tools in the market

To the best of our knowledge there is only one product in the market that offers spatio-temporal building localisation, named Mapillary. Contrary to their approach, we adapt to the user requirements of V4Design project and focus on specific classes of buildings of the considered dataset (Table 6). Comparison with this technique is presented in Section 7.3.3 providing a comparison of mIoU results for several stages of training phase in specific building classes.

7.3.3 Evaluation

In V4Design, the STBL module detects architectural structures, so we train our model for the aforementioned labels. The settings of the model involve a batch size of value 2, learning rate of 0.0001, momentum value equal to 0.9 and weight decay of 0.0005. In Table 7 we report the results of the computation of the mean IoU metric, using the Scikit-learn[1] library in Python.

Table 7: Mean IoU results for several stages of the training phase

Steps	mIoU
120,000	0.3658
140,000	0.3892
160,000	0.3680
180,000	0.3681
200,000	0.4539
220,000	**0.4602**

We observe that after 220,000 steps the mean IoU score reaches 46.02%, even at this challenging task of 18 total classes including the background class.

As an example, in Figure 17 and Figure 18 we present some qualitative results for the building localization in images and videos that content providers give us. In Figure 17, all buildings are segmented correctly, but in Figure 18 there are some examples in which the module was not be able to segment buildings correctly. Falsely segmented examples are usually very low-resolution images or images where the target building is covering only a small part of the image. Furthermore, in the case of images with low brightness, we also observe that the performance decreases significantly.

[1] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.jaccard_score.html

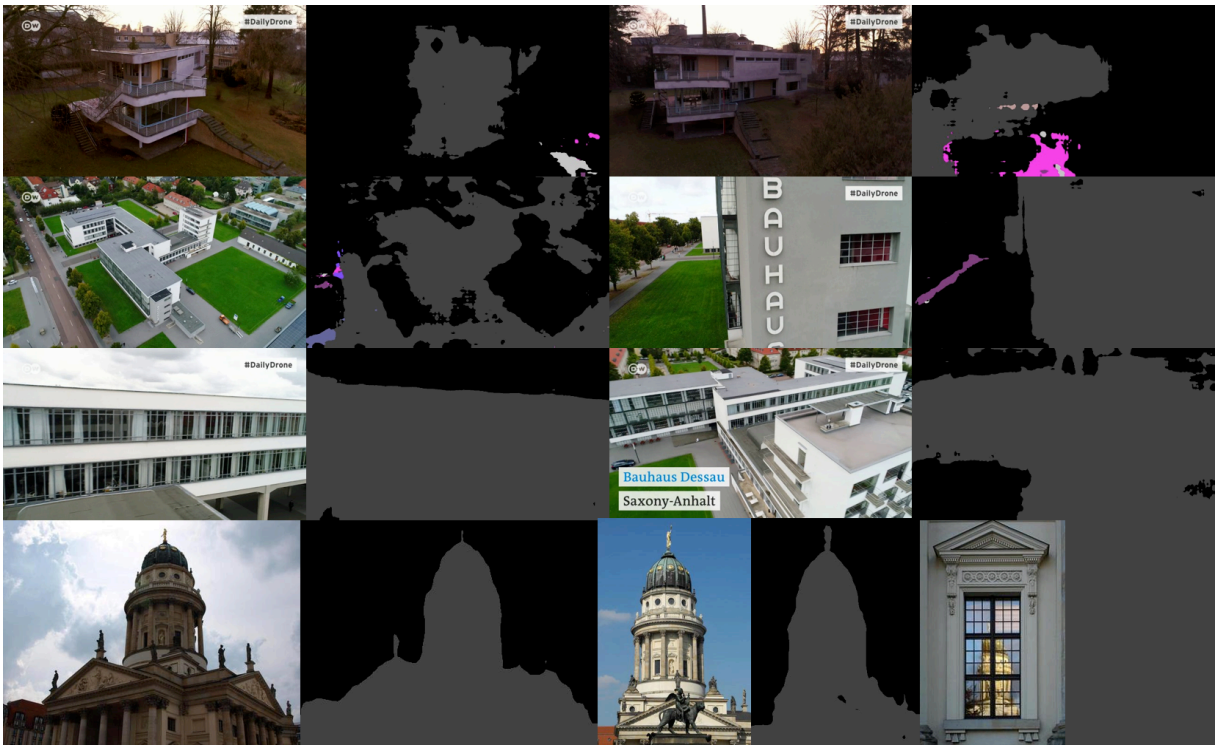


Figure 17: Truly Segmented images using V4Design STBL implemented approach

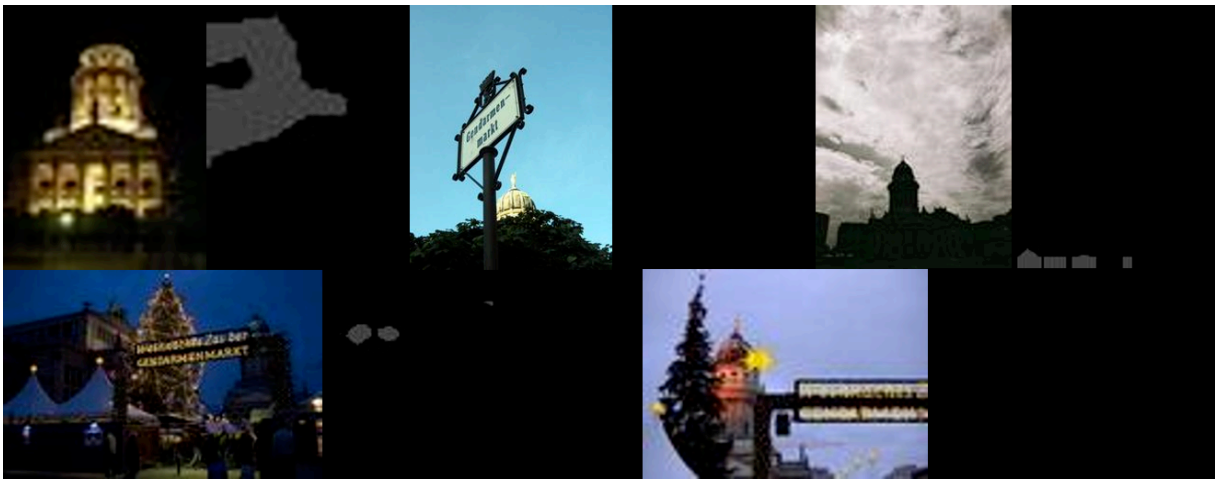


Figure 18: Falsely Segmented images using V4Design STBL implemented approach

8 CONCLUSIONS AND FUTURE WORK

8.1 Conclusions

Taking into account the analyses and results presented in this deliverable, we can conclude that all objectives and goals associated with user requirements (D7.2) have been successfully satisfied during the first period of the project (M1-M18). Initially, related work has been thoroughly studied and documented and then relevant architecture datasets have been accumulated and used in order to train the appropriate scene recognition and spatio-temporal building and object localization models. All relevant modules have been deployed, evaluated and tested in benchmark and V4Design visual data. Furthermore, they were also successfully integrated in the V4Design system. Technical requirements associated to each documented user requirement needs and satisfied them with the basic functionality outcome of the deployed modules.

8.2 Future work

As far as the future steps are concerned, that are envisioned to take place during the next implementation phase of T4.2, we foresee to introduce a spatio-temporal coherency throughout SR, STOL and STBL. We elaborate on this module in the following subsections.

8.2.1 Scene recognition (SR)

For Scene Recognition, we envisage that we could design and deploy a sophisticated algorithm that correlates SR predictions from time to time, so that it can maintain the temporal coherency amongst frames. In this way it will not produce false positive scene recognition predictions for neighbour video frame intervals and will diminish the flickering classification phenomenon between sequential video frames.

8.2.2 Spatio-Temporal Object Localization (STOL)

As far as STOL is concerned, we envision introducing a spatio-temporal coherency for the detected objects, so that we can monitor them throughout time. This could occur by deploying a spatio-temporal tracking of the detected bounding boxes and respectively the segmentation masks. In this way, the 3D reconstruction algorithm will be aware of the exact frames and intervals where a single object exists and will not make or require any assumptions from the reasoning component (T5.2).

8.2.3 Spatio-Temporal Building Localization (STBL)

Similar to the above components, we plan to expand STBL so as to include the coherency of its predictions throughout the temporal domain. In this way, it will be able to produce accurate and coherent pixel predictions for the building classes that it recognizes from frame to frame. The information will give the capability to 3D reconstruction to produce accurate and reliable labels to its computed 3D models.

9 REFERENCES

Cernekova, Z. a. (2005). Information theory-based shot cut/fade detection and video summarization. *IEEE Transactions on circuits and systems for video technology*, 82-91.

Chen, L.-C. a. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *pattern analysis and machine intelligence*, 834--848.

Gadde, R. a. (2017). Efficient 2D and 3D facade segmentation using auto-context. *pattern analysis and machine intelligence*, 1273--1280.

Gangopadhyay, A. a. (2016). Dynamic scene classification using convolutional neural networks. In *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)* (pp. 1255--1259). IEEE.

Girshick, R. (2015). Fast r-cnn. *international conference on computer vision* (pp. 1440-1448). IEEE.

Gong, Y. a. (2014). Multi-scale orderless pooling of deep convolutional activation features. In *European conference on computer vision* (pp. 392--407). Springer.

He, K. a. (2017). Mask r-cnn. *international conference on computer vision* (pp. 2961--2969). IEEE.

He, K. Z. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *pattern analysis and machine intelligence* (pp. 1904-1916). IEEE.

Huang, Y. a. (2019). Long-Short-Term Features for Dynamic Scene Classification. *IEEE Transactions on Circuits and Systems for Video Technology* (pp. 1038-1047). IEEE.

Jaderberg, M. S. (2015). Spatial transformer networks. *neural information processing systems*, (pp. 2017-2025).

Krizhevsky, A. S. (2012). Imagenet classification with deep convolutional neural networks. *In Advances in neural information processing systems*, 1097-1105.

Lin, G. a. (2017). Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. *computer vision and pattern recognition* (pp. 1925--1934). IEEE.

Lin, T. Y. (2017). Feature pyramid networks for object detection. *Computer Vision and Pattern Recognition* (pp. 2117-2125). IEEE.

Liu, H. a. (n.d.). DeepFacade: a deep learning approach to facade parsing.

Long, J. a. (2015). Fully convolutional networks for semantic segmentation. *computer vision and pattern recognition* (pp. 3431--3440). IEEE.

Martinovic, A. a. (2015). 3d all the way: Semantic segmentation of urban scenes from start to end in 3d. *Computer Vision and Pattern Recognition* (pp. 4456--4465). IEEE .

Neuhold, G. O. (2017). The mapillary vistas dataset for semantic understanding of street scenes. *International Conference on Computer Vision* (pp. 4990-4999). IEEE.

Patel, N. V. (1997). Video shot detection and characterization for video databases. *pattern recognition*, 583-592.

Peng, C. a. (2017). Large Kernel Matters--Improve Semantic Segmentation by Global Convolutional Network. *computer vision and pattern recognition* (pp. 4353--4361). IEEE .

Ren, S. H. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *In Advances in neural information processing systems*, 91-99.

Rosten, E. a. (2008). Faster and better: A machine learning approach to corner detection. *IEEE transactions on pattern analysis and machine intelligence*, 105-119.

Shalunts, G. a. (2011). Architectural style classification of building facade windows. *International Symposium on Visual Computing* (pp. 280--289). Springer.

Simonyan, K. a. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Tran, D. a. (2015). Learning spatiotemporal features with 3d convolutional networks. *international conference on computer vision* (pp. 4489--4497). IEEE.

Tsekeridou, S. a. (2001). Content-based video parsing and indexing based on audio-visual interaction. *IEEE transactions on circuits and systems for video technology*, 522-535.

Tsung-Yi Lin, M. M. (2014, September). Microsoft COCO: Common Objects in Context. *European conference on computer vision* (pp. 740-755). Springer.

Xie, S. G. (2017). Aggregated residual transformations for deep neural networks. *computer vision and pattern recognition* (pp. 1492-1500). IEEE.

Zeppelzauer, M. a. (2018). Automatic Prediction of Building Age from Photographs. *International Conference on Multimedia Retrieval* (pp. 126--134). ACM.

Zhou, B. L. (2017). Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 1452--1464.