Usage, Usability, and Utility of 3D City Models, 02011 (2012)
DOI: 10.1051/3u3d/201202011

Semantic alignment of documents with 3D city models

C. Tardy[1], L. Moccozet[2] and G. Falquet[1]

[1] *Institut d'Ingénierie des Connaissances et Logiques de l'Espace (ICLE), University of Geneva, Geneva, Switzerland*
[2] *Institute of Services Science (ISS), University of Geneva, Geneva, Switzerland*

Abstract. In urban semantic digital libraries, users need to access heterogeneous types of resources in order to achieve a given task. Contextualisation is known to increase the understanding of documents. With the use of 3D city models, we propose an alignment model that correlates, using the semantic annotations, the different resources of the repository with the different objects contained in the 3D model. The result of the application of the algorithm, will then allow the user to picture the relation of the documents with the city objects but also the relation they have between each other, as a whole, and with this contextualisation, increase its understanding of the resources.

1. INTRODUCTION

When performing a task that is related to a global urban issue, such as evaluating an urban project, a user needs to access several heterogeneous resources, in particular 3D city models (3DCM), text documents, images, videos, audios, scientific data sets, etc. Moreover, the user should be provided with tools to select relevant documents and with an interface that facilitates their visualisation and understanding. Designing such an interface is particularly challenging because 3DCM are generally better displayed inside 3D virtual environments while text documents (or images) are better represented in usual 2D windows. Moreover, the interface must represent the documents "in context", which means that a document referring to an object of a 3DCM must be visually linked to this object. Conversely, when navigating inside a city model one often needs to see documents that have a relationship with the currently visible city objects.

For a given document corpus and a 3DCM, this user interface generation problem can be decomposed into two steps: (1) Compute semantic alignment relationships between documents and objects in a 3D city model, *i.e.*, for each document determine the nature and strength of its relationship with each 3DCM object. (2) Use these correspondences to compute the user interface layout, *i.e.*, where and when documents appear within the 3D virtual environment.

In this paper we only focus on how to establish semantic alignment relationships between documents and 3DCMs. We also aim at solving this problem in the context of a semantic digital library of urban resources.

In a semantic digital library, every document is semantically indexed with a set of ontological elements such as concepts, relationships and axioms, drawn from one or more reference ontologies that form the annotation vocabulary. To create this vocabulary, we have aligned CityGML [1], Geonames [2], Urbamet [3] and other domain ontologies. This allows us to annotate the documents and the 3D models and to align them together. We present in this paper the alignment model and the algorithm that we have designed to create a matrix of the links between resources and 3D objects in the city model, as presented in sections 3 and 4.

This paper is organised as follows: In the next section we briefly summarise related works on the creation of interfaces and on alignment techniques to link documents and 3D environments. In section 3

we describe the semantic indexing model and the construction of the annotation vocabulary, in section 4 the algorithm and in section 5 two examples of its application. Finally, in section 6 we develop our conclusion and our perspectives for future work.

2. RELATED WORKS

Many researches have been done in the 3D visualisation of search results, such as Periscope [4], based on the Adaptive Visualisation Environments (AVE) method [5] and NIRVE [6]. The NIRVE[1] project displays the resulting set of documents of a search query from the NIST's PRISE [7] search engine. The cluster can be visualised on a 3D globe as a small box containing a bar chart representing the average concept profile of the documents. The latitude of the boxes on the globe reveals their relevancy and the thickness the number of documents they contain.

The AVE method proposes to automatically select the best visualisation based on the search results characteristics. The user can navigate, through the different visualisation methods, from a global view of the search result to a categorised view of sub-results. The Periscope visualization engine implements this method. It displays the results in a different 2D or 3D interface based on an index of readability.

The previously described works are focused on the representation of the returned information in a 3D environment. Those 3D information spaces are not contextualising documents according to their localisation in time and space. They create a space that is not related to the real world and that clusters the resources depending on different sets of information, according to their relevancy for the current user task. The urbanism domain needs a virtual interface that is closely related to our world and that will contextualise the returned documents according to their space, time and domain coverage. The time and space representation must closely match what the user is used to, as for example a map of existing cities. Furthermore, while the user interacts with the visualisation tool, the request to the search engine should evolve.

In urban planning, geographic information systems (GIS) are used to contextualise spatially the resources and the information. In GIS, the *geocoding* process consists in finding coordinates from textual data, such as street address, building name and area name [8, 9]. The problem we address here is similar, but instead of finding coordinates or map areas for a document, we seek to establish links between a document and 3D city model objects that are either directly referenced in the document or considered as relevant.

We can also cite the World Explorer project [10] that uses the Flickr photos database. The authors have developed a visualisation tool that shows the high-scored tags on a map of the world according to the zoom level. A user can then retrieve the pictures related to a tag and a place. This project uses localisation and concept information to retrieve geo-localised resources. Another interesting project, called VisGet, described in [11, 12], uses as World Explorer the localisation and keywords associated to a document. The user can navigate in a 2D map, select few keywords in a tag cloud but it can also define a time period to create the query to the system. The returned documents are then displayed as a list at the bottom of the interface.

Similar research have been done on linking documents and 3D models via ontologies, such as [13], where the authors have developed an ontology based model to align parts of documents or documents with a 3DCM, thanks to the creation of a dedicated ontology (OUPP), in the field of urban planning. Our approach is described in the following section.

3. CONTEXTUALISATION AND ALIGNMENT MODEL

As mentioned in the introduction, every resource (document, 3DCM, map...) is semantically indexed with a vocabulary obtained by aligning several ontologies, namely, Geonames, CityGML,

[1] Project home page: `http://www.itl.nist.gov/iaui/vvrg/cugini/uicd/nirve-home.html`

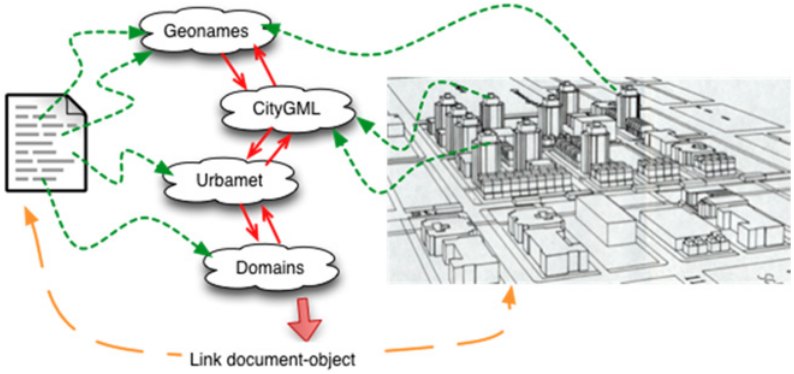

Figure 1. Contextualisation and alignment schema.

Urbamet, and specific domain ontologies, as depicted in Figure 1. The goal is to derive links between documents and 3D city model objects from their semantic index. In this section we show the construction principles and techniques we used to build the indexing vocabulary.

3.1 Resource indexing

The documents represent every non-geometric resource contained in the digital library. They can be decomposed in sub-documents that can be used independently.

Each document has a *coverage* that defines its time and space validity. It is composed of two groups of entities, one for each facet. In the present case, we only focus on the spatial aspect. The spatial coverage is composed of one or more entities from the Geonames ontology.

The content of a document is represented by a set of semantic annotations that link the document or some part of it, to entities of the annotation ontologies. Thus, an annotation can be a place name (*e.g.*, "Puerta del Sol"), a geographic/urban concept (*e.g.*, Lake, School) or a domain specific concept (*e.g.*, Infectious disease).

Similarly, the semantic annotation of a city object (*e.g.*, a building) is simply its representation in CityGML. Every 3DCM is represented by instances or subclasses of *CityObject* connected to instances of other CityGML classes. The coverage of a 3DCM as well as the coverage of each city object is given by their footprint (*GroundSurface*).

The coverage and the annotations entities will be used to define if there is a link between a city object and a document. The connections between them will be found thanks to the semantic vocabulary entities, as they are linked with each other in the annotation vocabulary.

A city object represents a 3D object in a 3DCM in CityGML format. [14]. As for a document, an object can be composed of other objects.

3.2 Aligning the ontologies

The Geonames ontology represents geographical name entities (*e.g.*, Places, Building) described with a name, an address, coordinates (one point in WGS84 format[2], describing the latitude, longitude and elevation) and feature codes. Feature codes describe the function of the entity (*e.g.*, School, River, and Park). We have transformed this ontology so that each *feature code* yields a class (*e.g.*, Monument) whose instances are the Geonames places with that code (*e.g.*, the Eiffel tower, Puerta del Sol). Each

[2] `http://earth-info.nga.mil/GandG/publications/tr8350.2/tr8350_2.html`

Figure 2. Ontology alignment.

geoname has a RDF[3] file attached, which describes its parents, its geographical hierarchy (*e.g.*, parent country, parent administrative division), but also the postal codes, the population, and more.

The CityGML ontology describes 3DCMs and landscapes, composed of city objects, with respect to their geometry, topology, semantics and appearance. We have created this ontology using the XML schemas provided in the specification [15]. The top classes represent the different types of city objects (*e.g.*, Land use, Building, City furniture). Each type has functions, which are described as subclasses. For example, the class *LandUseFunctionType* contains subclasses such as *Forest*, *Sea*, and *Airfield*. A city object has also a footprint that describes the polygon that marks the object sitting on the ground. For a building it is the *GroundSurface* class that contains those data. In the specifications, the address of an element can be composed of the postal address of the object, described in xAL format[4] and of the exact point of entry to the object.

The alignment of Geonames and CityGML creates an ontology of urban entities. The first one provides identification, information on the hierarchy between geographic features and their affiliation to the hierarchy of administrative divisions in a country. The later describes their geometry, their localisation, and defines city furniture such as benches or public lights. Finally, both give details on their function and this common attribute is the articulation of the alignment. For example, as described in Figure 2, the Geonames feature code *University* (S.UNIV) is aligned with the CityGML type *College_or_University,* which is a subclass of *BuildingFunctionType*.

To complete the annotation vocabulary, we have aligned different ontologies to represent the content of documents and to link them with this geographic ontology. We have chosen to use the first levels of the Urbamet thesaurus, as the main generic ontology, as it covers a wide range of domains but stays centred on urbanism. This thesaurus is filling the semantic gap between a concept and a city object or geographical name. For example, the concept *surgery* will be linked to the city objects that have for function *Hospital*. In Figure 2, the Urbamet concept *University* is linked with the Geonames feature code *UNIV* and the CityGML type class *College_or_University*. So it is aligned with the geographic and geometric ontologies previously described. Then all the other domain's ontologies that describe the different concepts, used in a particular expertise, are linked with this generic ontology, and through it, to the geographic ontology. This annotation ontology will index spatially and textually the documents and models stored in the digital library. It will then help users to formulate the queries but most of all, help in creating the links between the documents and the city objects to display them in the interface.

3.3 Aligning the individuals between CityGML and Geonames

3DCM objects are instances of CityGML classes while documents can be indexed with Geonames places (and other ontological entities). To establish semantic links between 3DCMs and documents, it

[3] `http://www.w3.org/RDF/`
[4] `https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ciq`

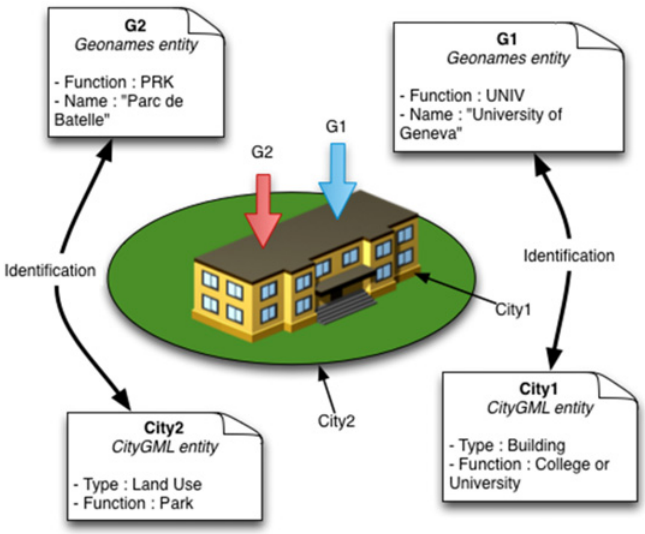

Figure 3. Identification of city objects.

is thus essential to match CityGML instances with Geonames places, i.e. to align individuals in addition to classes.

The alignment process is based on the following heuristics: (1) the place position (longitude, latitude) must be contained inside the 3D object footprint; (2) the place function must match the object's function. This can be tested because Geonames *feature codes* and CityGML *functionType* classes have been aligned. (3) Rules to solve ambiguities, when two or more 3DCM objects satisfy conditions 1 and 2. For instance, a Geonames road may correspond to both a *TransportationComplex* and a *TrafficArea* (of this complex); in this case, the larger *TransportationComplex* will be chosen.

Example. In Figure 3, there are two city objects: a building and a land use. The building (*City1*) is described as a *College_or_University*, and the land (*City2*) as a *park*. There are also two geonames place marks: *G1* and *G2* that both point inside *City1*. As *City1* is contained in *City2*, and as *G1* is described as a university (*UNIV*) and *G2* as a park (*PRK*), we can deduce that *G1* defines *City1* and that *G2* defines *City2*. This double verification highly reduces the risk of mismatching. Once this identification is done, the algorithm can search for links between objects inside the 3DCM and all the resources from the semantic digital library as described in the following section.

In order to identify building storeys with geonames, as for the identification of building, the geonames must be contained in the footprint of the objects, and their elevation must match the one of the storey. A building with three storeys might have four geonames inside its footprint, three of them describing each floor and the last one describing the whole building. The "storeys geonames" are described as contained in the "building geoname" using the *part_of* or *contains* attributes of the geonames. In CityGML, the storeys are represented using the *CityObjectGroup* class with the following attributes: *class* has for value "building separation", *function* has for value "lodXStorey" where X is the level of detail number and *gml:name* is the name or the number of the floor. The group is then composed of rooms, doors... and is associated with a *BuildingPart* or a *Building*. We have created a class *Storey* in the CityGML ontology to represent those rules. The next step is to identify each storey with the proper geoname. Each building has four attributes: *storeyAboveGround* and *storeyBelowGround* that contain the number of floors below or above the ground, and *storeyHeightsAboveGround* and *storeyHeightsBelowGround* that contain the total height of all the floors below or above the ground. In order to know the average height of each storey we need to divide the total height of the storey or

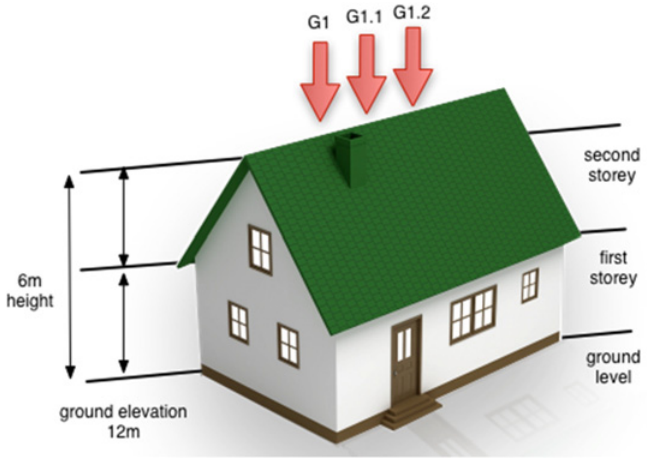

Figure 4. Identification of building's storeys.

above the ground with the corresponding number of floors. The ground elevation (g) added to the storey average height multiplied by its number (n), will give the appropriate elevation (e) of the geoname corresponding to the storey in question, above the ground. To calculate the underground elevation, g needs to be subtracted to the storey average height multiplied by $-n$, n being a negative number. We imply here that the ground floor is the storey zero; that the first floor is the first storey, *etc*. For the storeys above the ground, n has to be greater or equal to 0, and for the ones below, n is strictly less than 0.

$$g + \frac{storeyHeightsAboveGround}{storeyAboveGround} \times n = e \tag{1}$$

$$g - \frac{storeyHeightsBelowGround}{storeyBelowGround} \times (-n) = e \tag{2}$$

Example. As pictured in Figure 4, there are three geonames: *G1*, *G1.1*, *G1.2* that are all contained in the building *B* footprint. *G1* represent the building and has an elevation value of 12. *G1.1* and *G1.2* are both *part_ofG1* and they respectively have an elevation value of 12 and 15. *B* has a *storeyAboveGround* value of 2, and a *storeyHeightsAboveGround* value of 6. So each storey has an average height of 3 meters. *B1* is the first storey and *B2* is the second storey described as *CityObjectGroups*. The geoname attached to *B1* should have an elevation of $12 + (6 \div 2) \times 0 = 12$ according to formula 1, and be a *part_ofG1*. So *B1* is identified with *G1.1* and with the same methodology, we can say that *B2* is identified with *G1.2*.

4. AN ALGORITHM TO FIND THE ALIGNMENTS

This algorithm creates links between 3D objects and documents. The links can be of two types: *explicit* or *suggestion*. The first one describes a direct or indirect connection. If a document and a 3D object are indexed with the same entity (or with aligned entities), they are directly connected, but if they are indexed with different ones, there is an indirect link if there is an ontological connection between those entities. The second one can be used in two cases. First, it can represent a connection made using a non-geometric or non-geographic vocabulary. Finally, it can represent a link with an object that is not contained in the current 3DCM. However, the document is considered as relevant for the user and needs

to be cited in the results. So a link will be created with an object that is contained inside the 3DCM and that is close to the location of the appropriate one.

So the algorithm takes as input a document *doc*, a 3D city object *obj* from a 3DCM, both indexed with the indexing vocabulary. It will return as an output, a Boolean value for the *explicit_link* and a number value for the *suggested_link* that represent the weight of the suggestion link. This algorithm is using classes and individuals from ontologies. *Cov(doc|obj)* is the coverage of the document or the object. For the clarity of this explanation we are only treating the spatial aspect of the coverage in this algorithm, but a similar process has to be apply to the time facet. *Cov()* can only contain individuals or classes from the Geonames or the CityGML ontology. *Idx(doc)* is the list of concepts attached to a document to define its content. It can contain individuals or classes from any of the vocabulary's ontologies. *Idx(obj)* is the list of instances of CityGML classes that represent the object.

First of all the algorithm needs to check that the coverage of the document matches the coverage of the object, otherwise no link can be created. So *cov(obj)* needs to be included in or equal to *cov(doc)*, or vice versa. If this assertion is true, we can search for a link via the *searchLink(doc,obj)* function described below.

There are four possible cases where a document can be linked with an object. If the entities contained in both indexing list, they are identically or hierarchically linked as described in the cases 1 and 2. If the indexing terms for the document are linked to the function of an object, as explained in the case 4. Finally, as described in the case 5, when the object is not present in the model but the document is relevant and needs to be returned to the user.

1. Direct link: If *cov(doc)* and *cov(obj)* contain the same entity, then a link can be created between *doc* and *obj*, so *explicit_link* = true.

Example. If *cov(doc)* and *cov(obj)* both contain the "Geneva airport" geoname, so we can create an explicit link between the two entities.

2. Direct link by class: If *cov(doc)* contains a class c, and *cov(obj)* an individual e. If e is an individual of the class c, then a link can be created between *doc* and *obj*, so *explicit_link* = true.

Example. If *cov(doc)* contains the *airport* class from CityGML and *cov(obj)* the "Geneva airport" geoname. This individual is linked with the *airport* class, so we can create an explicit link between the two entities.

3. Link through non-geographic entities: g is an individual or a class, from one of the domain ontologies (*i.e.*, not form Geonames or CityGML). If *idx(doc)* contains g, *cov(obj)* contains f, and there exist an alignment in the ontologies between f and g, then *suggested_link* = 1/*semantic_distance(g, f)*. The semantic distance can be computed according to one of the well-known concept distance (or similarity) measures that have been defined for ontologies.

Example. If *cov(doc)* contains the *surgery* concept and *cov(obj)* the "Geneva university hospital (HUG)" geoname. This concept is linked with the *hospital* class, and "HUG" is a hospital. We can then create a suggestion link with the value of the semantic distance between the concept *surgery* and the geoname "HUG".

The previous cases can be in a particular situation if the object is known to be a group. So if *cov(doc)* and *cov(obj)* contains the individuals e and f, and if f is part of e (*i.e.*, if e is a building, f is a building part of e), then there are two possibilities. (1) The group is defined in the model and then we can identify the different objects individually. (2) The sub or top objects are not defined and we have to assign to the same object different geonames. For example, when a building contains offices that are individually identified in the Geonames ontology, but the 3D object is only defined as one single building. The algorithm has to test if e is defined as a group in the 3DCM, by calling *isGroupCityObject(obj,e,f,doc)* as described in Figure 5. This function will then call *searchLink(doc,obj)* with the new identification and fall back on the cases 1, 2 and 3. In order to know if an object is a group of city objects, it calls the *isAGroup(obj)* Boolean function. To verify if an object has been defined with a geoname entity, and to retrieve an object from a geoname that identifies it, it respectively uses *isDefined(obj,e)* and *getObj(e)*.

```
IsGroupCityObject (obj1: cityObject, e: individual, f: individual, doc: document){
      // search children in group
      if (isDefined(obj1, e) and f in e){
            if (isAGroup(obj1)){
                  obj2 = getObj(f)
                  searchLink(doc,obj1) and searchLink(doc,obj2)
            } else if (!isAGroup(obj1)){
                  // We define all the known individuals being part of "e" on the same
top object (obj1)
                  defineObject(obj1, f) and searchLink(doc,obj1) }
      }//search parent in group
      else if (isDefined(obj1, e) and e in f){
            if (∃ obj2 = getObj(f))
                  searchLink(doc,obj1) and searchLink(doc,obj2)
            else if (! ∃ obj2 = getObj(f)){
                  // Create obj2 as the union of its children. obj2 will not be a proper 3D
object in the scene, but a visual effect (colored zone, ...)
                  searchLink(doc,obj1) and searchLink(doc,obj2) }
      } }
```

Figure 5. IsGroupObject algorithm.

To assign to an object a geoname entity it calls *defineObject(obj,e)*. This function will associate a specific entity to a city object.

Example. If *cov(doc)* and *cov(obj)* both contain the geonames *g1* that represents a building that hosts many different companies, and *g2* that represents a part of this building owned by one of the enterprises. *obj* is defined by *g1*. The algorithm calls the function *isGroupCityObject*. We know that *g1 contains g2* in the Geonames ontology, but *obj* is represented as a simple building with no building part in CityGML. The function *isAGroup(obj)* will then return false. So it defines *g2* to be attached to *obj* as *g1*. Then it calls *searchLink* again to check if there is any document related to *g2*, and if there are, it will link them with *obj*. In the case where *obj* would have been defined with building parts, the function *isAGroup(obj)* would have returned true. Then the following steps would have been to retrieve *obj1* as the proper building part matching *g2*, to associate them, and to run the *searchLink(doc,obj2)*.

The last case is particular; in order not to miss relevant resources to be displayed to the user because the 3D object, they should be linked to, is outside the current 3DCM. We introduce here the coverage of the current 3D model as *cov(model)*. Its behaviour is identical to *cov(obj)*.

4. Link by spatial proximity: If *cov(doc)* contains an individual e, and *cov(model)* an individual f. If e is geographically close to f, then a suggestion link can be created between *doc* and *obj*, where *obj* is an object from the model and is spatially positioned next to e. The *suggested_link* is equal to the addition of the *suggested_link* value from the case 3, and 1/*euclidean_distance(e,f)*.

Example. If *cov(model)* contains the geoname (f) "Lyon 3rd" which is the 3rddistrict of the city of Lyon in France. Cov(doc) contains the geoname (e) "Lyon 6th"which is the 6th district of Lyon. e and f are spatially close, as they have a border in common. The algorithm will create a suggestion link between *doc* and an object *obj* contained in f that is close to the border between f and e.

This algorithm generates a matrix M, where each column corresponds to an object in a 3DCM and each line represents a document stored in the library. If there is a link between a document and a city object, then *M(doc,obj)* will contain the link value: *explicit_link* or *suggested_link*.

5. EXAMPLES

To illustrate the principle of the algorithm, we detail below a first example of identification and link through non-geometric entities.

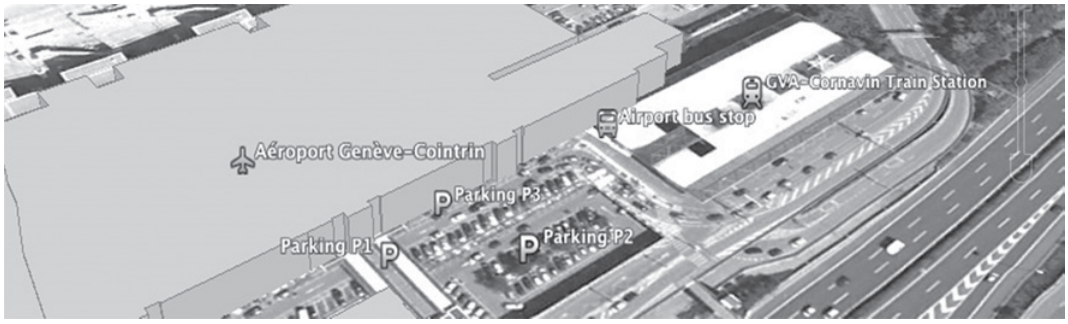

Figure 6. Geneva international airport.

doc is an article from the "Journal de l'aviation"[5] explaining that Swiss airline is now equipping its head crew members with digital tablets. The document is annotated with different concepts, such as *Swiss*, *airline*, *tablets* and *crew*. *Cov(doc)* is the geoname that represents the whole Switzerland country (geonameId: 2658434).

We use in this example the 3DCM of Geneva Canton. It contains *obj,* which is a 3D building representing the Geneva airport and described with a *BuildingFunctionType* with the value *Airport_Building*. *e* is an individual of the Geonames ontology: {label: "*Aéroport Genève Cointrin*", geonameId: 2660644, feature code: *Airport*(AIRP)}. In the annotation ontology, *AIRP* is aligned with *Airport_Building*, and *e* is positioned inside the object footprint. So *obj* is identified with *e*.

Next step is to find if there is a link between the document and an object in the model. $Cov(doc) = s$ and this geoname contains the 3DCM coverage (Geneva Canton). This implies that a link is possible with an object in the model. *Idx(doc)* contains "airline" which is linked with the concept "airport" in Urbamet. In the ontology, this Urbamet concept is linked with the Geonames feature code *AIRP*. So there is a link between *doc* and all the objects in the model that are defined as airports, here *obj*. Finally, *M(doc,obj)* contains $explicit\_link = \text{true}$.

In Figure 6, there are six geonames represented by the different icons. *doc* is associated with the object that is defined by the geoname with the plane icon.

This next example illustrates the cases where the algorithm needs to use objects that are not identified with geonames, but only with CityGML classes, in the model.

doc is the contact webpage of the Human Resources of the University of Geneva[6]. They specify that their offices are located in the first floor of the "UniDufour" Building in Geneva. So *cov(doc)* is composed first of a geoname *g*, that represents the "UniDufour" building of the University of Geneva (geonameId: 8285539), and of a class *Storey*, with the specified value for the *name* attribute "1". The coverage is the element of the building "UniDufour" that is of class *Storey* and is located in the first floor.

obj is the 3D representation of the building "UniDufour" in Geneva, and *cov(obj)*=*g*. This object is a group and is composed of four storeys above the ground. The 3D objects *o1*, *o2*, *o3* and *o4* represent those four different building parts. They have their *name* attribute set to the floor level they represent.

Cov(doc) and *cov(obj)* both contain the same geoname *g*. So they can be associated, but *cov(doc)* contains also a specification, which is the class *Storey* with the value of the *name* attribute set to "1". As obj is a group we can run the *searchLink* function with each of its sub-object and try to find a match

[5] Can be accessed at this address: `http://www.journal-aviation.com/actualites/17176-le-metier-de-chef-de-cabine-se-numerise-chez-swiss`

[6] Can be accessed at this address: `http://www.unige.ch/adm/dirh/contact.html`

with *doc*. *Cov(o2)* matches this requirements, so we can complete the matrix with *M(doc,o2)* containing *explicit_link* = true.

6. CONCLUSION AND FUTURE WORK

In this paper we have proposed a technique to compute semantic alignment between documents and city objects of a 3D city model. The main advantage of this technique is that it entirely relies on the semantic indexing of documents and 3DCM. So, the indexing of a document is independent of the 3D model in which it will be displayed. Unlike the other researches presented in section 2, that only search for direct explicit links between resources and the query terms, we have further develop the computations of the link between the resources and the 3D objects. The algorithm is capable of identifying links over different degrees of connections in the vocabulary. The proposed algorithm must still be evaluated by comparing the computed results to correspondences established by a human expert. For this purpose, we are currently creating a realistic use case.

This algorithm doesn't yet properly take into account fuzzy defined areas, such as the city centre. Many research have been published [16], to define the exact zone, but the concept is very subjective to everyone. We consider this kind of area as the union of all the objects attached to a document that is tagged with such a concept and that is only associated with a unique object in the model. We then fall back in our non-geographic link case. This question will be developed in further research.

These correspondences, together with the 3D city model and the documents form a semantically rich model that can be viewed through different visualization techniques. In further work we plan to develop the different visualisation techniques, and the query system, in order for the user to visualise inside an immersive 3D environment of a city the resources that are relevant for him. In order to develop the search engine, we plan to combine information from the user profile, the user current task, and some concepts from the annotation vocabulary to refine the query. The user profile is composed of a coverage (space and time) and a domain of interest. The task a user performs could be to find a suitable place for an urban solar power plant. Such task implies certain parameters (concepts, location, time and date...) to be given to the search engine. After a query has been submitted, it will return a 3DCM and documents according to the link matrix, to create an interface. In order to refine the query, the level of detail, the zoom level and the position of the user in the model will be taken into account.

References

[1] CityGML. *http://www.citygml.org*. Accessed: 16-06-2012

[2] GeoNames Ontology. *http://www.geonames.org/ontology/documentation.html*. Accessed: 16-06-2012

[3] Urbamet. *http://www.urbamet.com*. Accessed: 16-06-2012

[4] W. Wiza, K. Walczak, W. Cellary, *Proc. of the 9th Int. Conf. on 3D Web Technology.* Periscope - A System for Adaptive 3D Visualization of Search Results. 29–40 (2004)

[5] W. Wiza, K. Walczak, W. Cellary, *IADIS Int. Conf. e-Society*. Adaptive 3D interfaces for search result visualization. 365–372 (2003)

[6] J. Cugini, S. Laskowski, M. Sebrechts, *Proc. of IST/SPIE's 12th Annual Int. Symposium: Electronic Imaging*. Design of 3D visualization of search results: Evolution and evaluation. 23–28 (2000)

[7] PRISE Search Design Notes. *http://www-nlpir.nist.gov/works/papers/zp2/psearch_design.html*. Accessed: 16-06-2012

[8] Y.Y. Chen, T. Suel, A. Markowetz, *Proc. of the ACM SIGMOD Int. Conf. on Management of data*. Efficient query processing in geographic web search engines. 277–88 (2006)

[9] C.B. Jones, A.I. Abdelmoty, D. Finch, G. Fu, S. Vaid, *Proc. of the 3rd Int. Conf. on Geographic Information Science*. The SPIRIT Spatial Search Engine: Architecture, Ontologies and Spatial Indexing. 125–139 (2004)

[10] S. Ahern, M. Naaman, R. Nair, J.H.I. Yang, *Proc. of the 7th JCDL*. World explorer: visualizing aggregate data from unstructured text in geo-referenced collections. 1–10 (2007)

[11] M. Doerk, S. Carpendale, C. Collins, C. Williamson, VisGets: Coordinated Visualizations for Web-based Information Exploration and Discovery. *IEEE Transactions on Visualization and Computer Graphics*. **14**. 6. 1205–1212 (2008)

[12] M. Dörk, C. Williamson, S. Carpendale, 18th *Int. WWW Conf.* Towards visual web search: Interactive query formulation and search result visualization (2009)

[13] C. Metral, G. Falquet, M. Vonlanthen, *Ontologies for Urban Development*. An ontology-based model for urban planning communication. 61–72 (2007)

[14] T.H. Kolbe, G. Gröger, L. Plümer, *1st Int. Symposium on Geo-Information for Disaster Management*. CityGML–Interoperable access to 3D city models (2005)

[15] G. Groger, T.H. Kolbe, A. Czerwinski, C. Nagel, *OpenGIS®City Geography Markup Language (CityGML) Encoding Standard*. Technical Report #OGC 08-007r1 (2008)

[16] P. Lüscher, R. Weibel, *Proc. of the 13thWorkshop of the ICA Commission on Generalisation and Multiple Representation.* Semantics matters: Cognitively plausible delineation of city centres from Point of Interest data (2010)