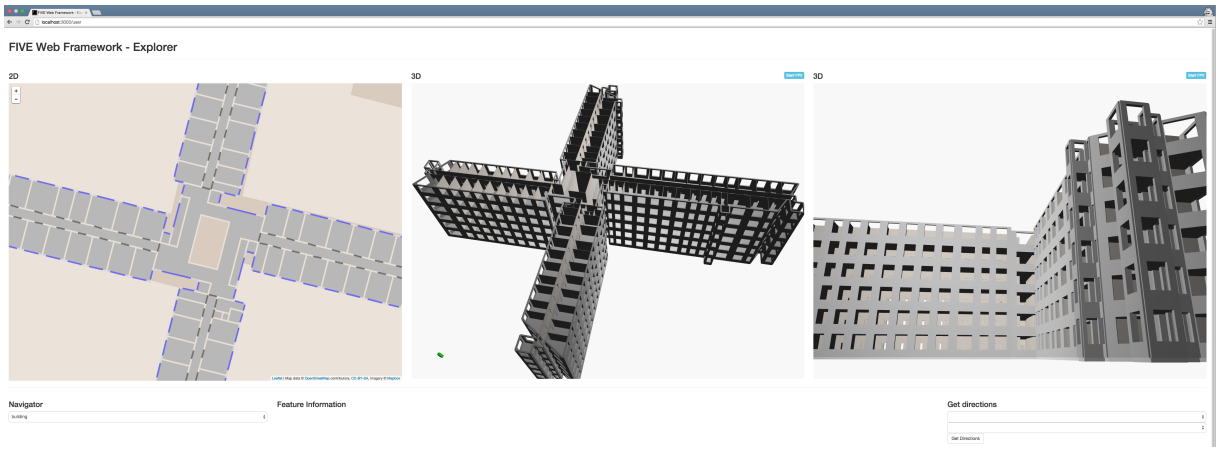
STAG: Smart Tools & Apps for Graphics (2015)
Silvia Biasotti and Marco Tarini and Andrea Giachetti (Editors)

HIJSON: cartographic document for web modeling of Interactive Indoor Mapping

F. Spini[1], M. Sportillo[1], M. Virgadamo[1], E. Marino[1], A. Bottaro [2] and A. Paoluzzi[3]

[1]Dipartimento di Ingegneria, Università Roma Tre, Rome, Italy
[2]Sogei S.p.A., Ricerca e Sviluppo, Rome, Italy
[3]Dipartimento di Matematica e Fisica, Università Roma Tre, Rome, Italy



Abstract

This paper introduces HIJSON, a novel indoor cartographic document format with several enhancements. The document is generated using LAR (Linear Algebraic Representation), so allowing for cellular complexes with general topology and shape. A software framework FIVE is also presented, that relies on HIJSON documents and is entirely based on web technologies. With respect to current cartographic formats, HIJSON suggests four major enhancements: (a) exposes a hierarchical structure; (b) uses local metric coordinate systems; (c) may import external geometric models; (d) accepts semantic extensions. The semantic extensions supported by the FIVE framework architecture encapsulate the details about communication protocols, rendering style, and exchanged and displayed information, allowing the HIJSON format to be extended with any sort of models of objects, sensors or behaviors.

1. Introduction[†]

For environments with massive presence of sensor-equipped (or "smart") objects, which realize the IoT (Internet of Things), an *Interactive Indoor Mapping* (IIM) system represents the ideal interface for integrated IoT monitoring. To be specific, IIM provides the container of indoor navigation systems, giving also the user, routed across an indoor environment, the opportunity to interact with objects along the suggested paths. Furthermore, it exploits the advancements in the field of indoor user's radio frequency identification and location, whose efforts are nowadays focused

[†] This work was partially supported by grant 2014/15 from Sogei S.p.A., the ICT company of the Italian Ministry of Economy and Finance.

© The Eurographics Association 2015.

to realize an integration of positioning systems like GNSS (Global Navigation Satellite system), Wi-Fi, Bluetooth and LTE (Long Term Evolution). By supporting continuous outdoor/indoor navigation by means of integration of technologies, an IIM system represents the most natural interface to perform realtime access monitoring and multi-person tracking. Self-supporting web-based technologies and context-aware objects, in conjunction with location systems, represent the enabling components for the scenario (depicted in Figure 1) we are relentlessly moving toward: the *Web of Things*.

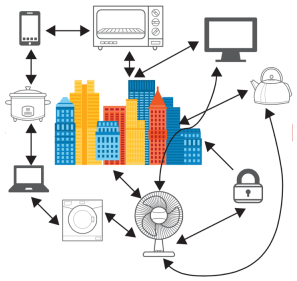


Figure 1: Web of Things (2010–2020?). Image from [PSB15].

An interactive mapping platform allows the representation of the indoor environment of both public or commercial places of vast dimensions, as for example airports, train stations, shopping malls, and also private buildings subject to strict access protocols, like warehouses, logistic centers, data centers, etc. Despite of the growing attention regarding indoor cartography, efforts to specify open formats for indoor representation are few and partial, and certainly not intended to support the interactive indoor mapping, which is conversely the main purpose of this paper.

The work, jointly developed with Sogei S.p.A., an ICT company fully owned by Italian Ministry of Economy and Finance, and the CVDLAB (Computational Visual Design Laboratory) of the "Roma Tre" University, is inspired by the needs of Sogei itself, which runs one of the largest data center of Europe, so requiring very strict access control policies, which include the recording and the real-time interaction with man/machine maintenance scenarios. Support for this interactive framework, where realtime awareness of the maintainer position inside the data center helps to reduce intervention times and to increase safety and security, has been chosen as case study of interactive indoor mapping, based on the proposed indoor cartographic format.

Research on the cartographic representation of indoor environments is extensive and heterogeneous with respect to the strategies applied [GGM12]. Different information sources are used, and accuracy of the produced solution depends on the adopted approach. In some cases the information is obtained with automatic or semi-automatic processing of files that describe the architectural structure of a building, such as BIM (Building Information Modeling) [ETSL08] and/or IFC (Industry Foundation Classes) that describe a building project [BDHL*14]. The actual "de-facto" standard in terms of geospatial data representation is the GeoJSON format [BDD*08], which can be easily used for any type of geographical annotation. In some cases it has been slightly adapted to be used in indoor environments: it is the case of the IndoorJSON project [ind13].

1.1. Our contribution

The focus of this work is the definition of a novel format of cartographic documents along with the software ecosystem rooted on it. HIJSON (Hierarchical Indoor JSON) is the name chosen for the new document format. Along with the accompanying software framework (https://github.com/cvdlab/hijson), it aims to realize a mapping between the real indoor spaces and a virtual interactive web environment. A simple but effective algorithm to find indoor valid routes is also provided.

Hierarchical structure

The HIJSON format allows for hierarchical description of indoor spaces, reflecting a container-contained relationship. This directly implies a neater representation then the plain linear structure adopted by GeoJSON, being a perfect analogy of objects contained (i.e. placed) into spaces. Therefore, an organized arrangement of spaces is allowed by HIJSON, via logical (or even physical) grouping: concepts like building wings, sections, storeys, departments, etc. can be directly introduced, in order to reflect into the document structure the actual logical or physical divisions, categories or relationships among the modelled spaces. Furthermore, the container-contained relation enables a recurring use of local reference frames.

Metric local coordinate system

Supported by the hierarchical underlying structure, the HIJSON document format allows the use of local coordinate systems. Hence the shape of all elements can be conveniently modelled using local coordinates, and then placed in the right position with respect to the position of the parent (or container) element applying a rotation, followed by a translation transformation. Another substantial advantage is represented by the adoption of a metric reference frame, consequently simplifying the compilation of the document, either manually generated or produced by software tools. Just remember that the GeoJSON coordinates are geographical, a pairs of (absolute) latitude and longitude angles, like the ones provided by GNSS systems. This kind of coordinates are certainly not particularly user friendly, when positioning a smart device or a furniture element within a specific building room.

© The Eurographics Association 2015.

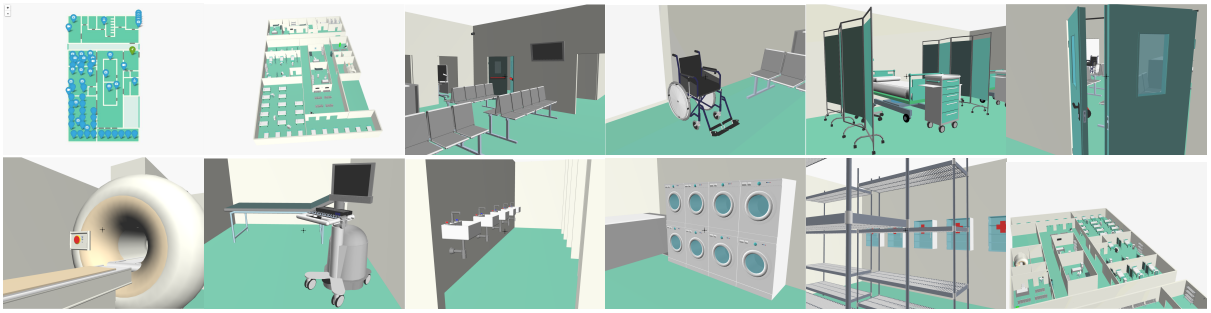


Figure 2: HIJSON visualization examples in the Emergency Department of a general hospital model. The 3D visualization includes furniture and smart devices.

Hyperlinked geometric models

The HIJSON document may further import external geometric models, either of the buildings themselves or the interior furniture or devices, that are topologically complete (in the sense of solid modeling [15]) and very compact. Such models, coming from a source outside the document, are acquired by hyperlinking JSON files that contain a Linear Algebraic Representation (LAR) of topology and geometry, to be expanded for visualization or interaction at any useful level of detail (see figure 3). The LAR scheme [DPS14] is characterized by a very large domain, including architecture, building and construction [PMS15], 2D and 3D engineering meshes, non- manifold geometric and solid models and meshes, and high-resolution 3D images [PDFJ15]. The expansion of a LAR model, to be considered as a general-purpose graphic primitive, may be executed on either the server or the Supervisor client of the FIVE Web Framework (see Section 4.1), or even on the Explorer client.

Semantic extensions

Semantic extensions make the HIJSON format extendible and customizable, that is able to adequately respond to any need of objects representation. To define a semantic extension means to allow the HIJSON document to model an object previously not covered, or even to modify the behavior of a comprised one. Semantic extensions are to be defined both as HIJSON format syntax and as HIJSON Toolkit source code. In particular it is necessary to define respectively a new HIJSON Element and a new HIJSON Class.

2. Indoor Modeling of Buildings

The geometric modeling of interior spaces of complex buildings and architectural environments, with the aim of producing a textual document, aimed to be processed by common web mapping tools, is not a straightforward task. Actually it requires a sequence of modeling steps that demand for different topological and geometrical concepts, introduced and briefly discussed in this section.

2.1. Background

Architectural modeling

The *architecture* of a *building* is characterized by two interacting and mutually dependent *systems*, respectively relative to (a) the interior/exterior spaces where the humans live and perform their activities and to (b) the manufacts that realize the physical covering of interior spaces and provide the boundaries of exterior spaces. In short, when modeling an urban area including a set of buildings and their indoor spaces, we must consider both the system of *building spaces* and the system of *building components* [CP80,CK94, Ger96].

The system of building spaces may be characterized as a partially ordered family B (Building Units) of subsets of elementary spaces S (Space Units). The set S of space units provide a *partition* of the *Building Space*. The set B of building units is a covering, partially ordered by containment, of the Building space providing a hierarchical structure to it.

The system of building components, also called *building fabric*, can be partitioned into the *building envelope* and the *building partitions*. The *building envelope* is the physical separator between the conditioned and unconditioned environment of a building, including the resistance to air, water, heat, light, and noise transfer, along to give buildings form, and to provide shelter and security. The *building partitions*, either horizontal or vertical, enclose the indoor spaces, and are contained within the building envelope. The former are roofs, floors and ceilings; the latter aim to support the horizontal partitions, either massively or by means of a building frame, where the loads are transferred.

Cellular complexes and sparse matrices

Therefore, to model the indoor space we need to partition the building into *cells* corresponding to the unit spaces, aggregate them to assembly the hierarchical building units into *cell complexes*, take control of the adjacency relations though suitable *topological operators*, extract the *boundary chains* of either 2D or 3D *chains* of cells, in order to generate the various families of building components [PMS15],

© The Eurographics Association 2015.

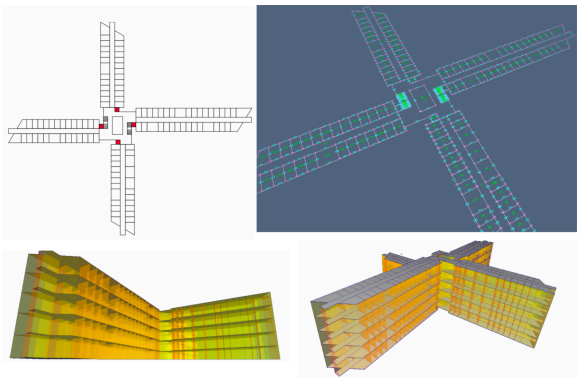


Figure 3: Office building: (a) the schematic plan —made of SVG primitives: lines, rects, polygons— possibly obtained via a client UI loading engineering design in background; (b) the automatically generated LAR cellular complex, transformed server-side into HIJSON; (c and d) the mock-up for the automatically generated 3D HIJSON.

and extract the *coboundary chains* of subsets of 1D cells or 2D cells, to compute the graph of all paths within the building.

A *cell complex* X is a partition of a compact space $S \subset \mathbb{E}^n$ into compact subsets of points, called *cells*, such that the intersection of every pair of cells is either empty or is a *boundary face* of both [Hat02, Ghr14]. The cells have a *dimension*. 0-cells are topologically equivalent (homeomorphic) to single points (vertices), 1-cells to curves (edges), 2-cells to surfaces (faces) and 3-cells to solids. The set of d-cells is denoted as X_d, so that X can be seen as a *stratification* $(X_0, X_1, \ldots, X_d)$. With abuse of language we call X_h the h-skeleton of X, and $K_h : \{1,2,\ldots,n_h\} \to X_h$ the *indexing maps* of h-cells $(0 \leq h \leq d)$, where $n_h = |X_h|$.

When the X_0 skeleton is embedded into the Euclidean space $\mathbb{E}^d$, providing a one-to-one mapping $\mathtt{V} : K_0 \to \mathbb{E}^d$ of 0-cells with the *vertices* array, a piecewise linear shape may be associated to a h-cell λ via its characteristic function as a subset of $\mathtt{V}$. Let us remember that given a subset A of a larger set, the characteristic function $\chi(A)$ is defined to be identically one on A, and zero elsewhere.

The cellular complex X can be represented combinatorially by $d+1$ families of characteristic functions of h-cells as subsets of vertices. For the sake of readability, let us use the symbols $\mathtt{V}$, $\mathtt{E}$, $\mathtt{F}$, $\mathtt{C}$ for K_0, K_1, K_2, and K_3, respectively, and $\mathtt{EV}$, $\mathtt{FV}$, $\mathtt{CV}$ for the binary matrices to be read as "edges by vertices", "faces by vertices", and "cells by vertices", having as rows the images of characteristic functions of 1-, 2- and 3- cells as discrete sets of vertices.

It is well known that subsets of h-cells in a cellular complex can be seen as elements of a linear space C_h of *h-chains* over the finite field $Z_2 = \{0,1\}$. When the indexing of cells (i.e. their ordering) is fixed, the *characteristic matrices* $\mathtt{EV}$, $\mathtt{FV}$, $\mathtt{CV}$, that can be seen as maps from cells to vertices, provide the bases for C_1, C_2, and C_3, respectively.

The boundary operators $\partial_3 : C_3 \to C_2$ and $\partial_2 : C_2 \to C_1$, computable by transposition, multiplication and filtering of values of two characteristic matrices [DPS14], provide the basic tool to study the topology of a cellular complex. The product of the ∂_d matrix times the characteristic vector of a d-chain returns the *boundary* $(d-1)$*-chain* of facets exposed on the border of such subset of d-cells.

Of course, characteristic and boundary matrices are *very sparse*, with sparsity growing fast with the number of cells, since the number of vertices for cell is small and bounded. Therefore, a list of lists of integers suffices to represent such matrices, and special storage methods [BG12], and in particular the `CSR` (compressed sparse row) and `CSC` (compressed sparse column), will provide fast computation of the boundary facets of *every subset of cells*.

LAR: Linear Algebraic Representation

A *representation scheme* is a mapping between the mathematical spaces to be represented by a computer system and their symbolic representation in computer memory. The *Linear Algebraic Representation* (LAR) scheme [DPS14], was recently introduced for topological methods to represent and process mesh connectivity information for dimension-independent cellular complexes, including simplicial, cubical, polytopal, and more general (non-convex and non simply-connected) cells.

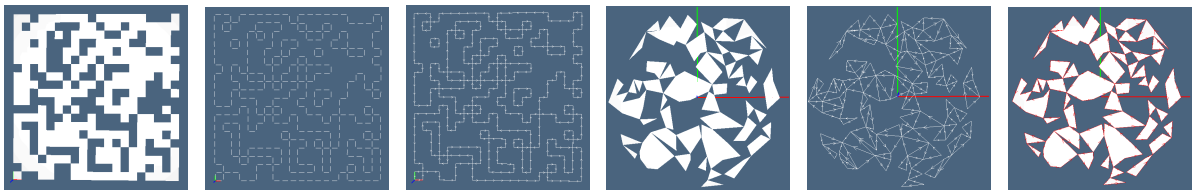
The Linear Algebraic Representation (LAR) scheme uses Combinatorial Cellular Complexes (CCC) as mathematical domain [Bas10] and various compressed representations of sparse matrices [WOV*07, BG12] as codomain. This scheme works even with the complicated domain partitions—with non-convex and/or non-manifold cells, and cells non homeomorphic to balls—that may arise in Building Information Modeling (BIM) and Geographic Information Systems (GIS) applications.

Simplicial and cuboidal cell complexes provide the standard mesh representation used in most science and engineering simulations, whereas complexes of possibly non-convex or non-contractible cells may be needed to represent the built environment in software applications for the Architecture, Engineering, and Construction (AEC) sector [PMS15].

LAR implementation

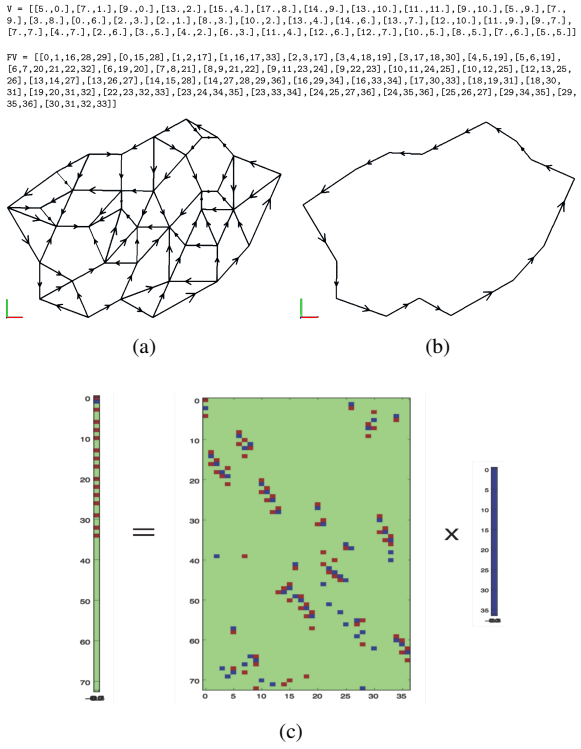
The architectural modeling discussed in this paper is written in Python, using the `Pyplasm` and `Lar-cc` libraries. The application produces `HIJSON` documents containing 2D plan models of building embedded in 3D (see Figure 7b). Those are mapped to complete 3D virtual models, visitable in first person from the client-side FIVE framework [Spo15, Vir15]. The `Pyplasm` library is a python in-

© The Eurographics Association 2015.



(a) (b) (c) (d) (e) (f)

Figure 4: LAR examples: (a) 2-chain of quad cells; (b) boundary 1-chain; (c) oriented boundary; (d) random 2-complex; (e) oriented 1-skeleton; (f) oriented boundary (in red).



(a) (b)

(c)

Figure 5: A toy example of the LAR scheme: (a) the bare minimum of data with *complete* information about topology; (b) the extracted boundary; (c) the extraction method $[e] = [\partial][f]$ giving the coordinate representation (in the discrete basis of the 1-cells) of the boundary edges $[e]$ by product of the sparse boundary operator matrix $[\partial]$ times the coordinate representation $[f]$ of the 2-cells (faces), in the discrete basis of the 2-cells.

terface towards a C++ geometric kernel, current implementation of the PLaSM language [PPV95, Pao03], developed as geometric extension of the functional language FL, by Backus and the functional programming group of IBM Research at Almaden [AWW91, BWW90, Bac78].

The Python prototypal implementation of `lar-cc` is in progress as an integral part of a permanent effort to rethink the foundations of solid modeling, by simplifying and generalizing its data representation and disentangling its main algorithms. The computational framework aims to accommodate the new world of geometric data over cloud- and web-based infrastructures. This long-term project has already achieved some tangible results in applications for the extraction of solid models from 3D medical images [PDFJ15] and to the simplified generation of building models for indoor mapping and IoT (as documented in this paper).

2.2. Hierarchical cellular complex

The linear chain spaces C_h ($0 \leq h \leq d$) can be enriched with a preorder structure, induced by subset containment, providing for a richer denotation of the family of subsets of cells, which corresponds to the set union of *BuildingUnits*, and *BuildingSpaces*. A corresponding set of names is also introduced, so that every substructure, at every hierarchical level, is associated to a unique ID, generated by concatenation of ancestor names along the (unique) path from the hierarchy root to the current node in the traversal of the hierarchy. Both the 2-cells and the 1-cells are inserted in the hierarchy. The former because of functional requirements, the latter to automatically select the building components where to automatically insert the openings. Doors and windows are so automatically associated to appropriate hierarchical 1-chains.

Boundary and coboundary operators

The boundary operators ∂_h are used for fast computation of the possibly unconnected or multiply connected horizontal partitions of the building, where necessary. Their dual *cobounbary* operators $\delta^{h-1} := \partial_h^\top$ are used to compute the chains of 2-cell sharing a common door, starting from the 1-chain corresponding to the 2-boundary of either horizontal or vertical communications, in order to automatically build the graph of all continuous paths inside the building.

Building structures and assemblies

Hierarchical models of complex assemblies are generated by aggregation of subassemblies, each one defined in a local coordinate system, and relocated by affine transformations of

© The Eurographics Association 2015.

coordinates. This operation is repeated hierarchically, with some subassemblies defined by aggregation of simpler parts, and so on, until one obtains a set of elementary components, which cannot be further decomposed.

Two main advantages can be found in a hierarchical modeling approach. Each elementary part and each assembly, at every hierarchical level, are defined independently from each other, using a local coordinate frame, suitably chosen to make its definition easier. Furthermore, only one copy of each component are stored in memory, and instanced in different locations and orientations how many times it is needed.

2.3. Production of HIJSON document

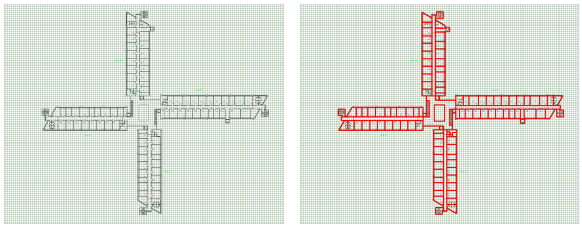
The RAD of building models is produced via a combination of server-side and client-side computing, based on text-based geometric computing and interactive data transformation via graphical input actions. Server-side, some input floorplan images are transformed into a 2.5D cellular complex, until to produce the textual HIJSON document, according to the GeoJSON format. Client-side, the HIJSON document is received via http, parsed and transformed into either a 2D map, or a 3D extruded solid model, or both, depending on the type of device and/or the UI status.

Within the transformation process, we therefore distinguish between:

(a) input of wire-frame drawings from architectural floorplan images into standard web representation of vector drawings, followed by parsing of textual (`.svg`) files;
(b) generation of a 2D cellular complex for long-term storage (`.lar`) and hierarchical modeling of cellular models using a very simple and general geometric format (LAR) based on algebraic topology methods;
(c) transformation of the `.lar` format into a structured 2D description providing semantics to spaces and possibly to the different elements of the building fabric, by using an object-oriented hierarchical interactive description hinged about the concept of `Struct` instance, to be not confused with the structural backbone of the building.
(d) Such structured and semantically annotated building model is finally exported as a rich textual 2.5-dimensional description using the `HIJSON` format.
(e) The `.json` files are finally transformed client-side into both 2D and 3D environments allowing the realtime spatial placement and user-tracking within the virtual environment of the individuals moving inside the real building.

It is worth noting that the whole construction of the 2.5D model—i.e. the cellular 2-complex embedded in 3D—is made using normalized local coordinates, where the root coordinates are contained in the $[0,1]^3$ interval. The transformation to geographical cylindrical coordinates is executed at generation time of the HIJSON document. To get the best numeric accuracy, the planar rototranslation component depends on three extreme points of the building plan, whereas the z-scaling needs the knowledge of the building height. The affine transform is propagated to each node using the semantics of the `Struct` primitive, analogous to the one of the old, good PHIGS graphics standard.

From raster images to vector drawings



(a) (b)

Figure 6: Input image: (a) `.png` input file of a typical floorplan of a real business tower; (b) user-interpretation of plans as wire-frame drawings (in red).

The architectural shape is introduced in the system using a rasterization of its plan drawings. In the test-case[‡] we used a `.png` raster file of the first building floor, and supposed the layout constant on the various storeys. Of course, the number of input files will correspond to the number of different floor plans. In Figure 6a an image is shown of the input image.

An user-interpretation of plans as wire-frame drawings is performed (see Figure 6b), by using some interactive drawing application, so producing one or more `.svg` files, that can be written and read by most interactive drawing applications, both commercial and open-source. Such files can be also searched, indexed, scripted, compressed, and scripted using JavaScript. The mechanism of *layers*, if available, is useful to have the building plans correspond exactly. A metric scale is not fixed in advance, since the geometric representation of the building will be subsequently transformed to any useful scale, position and orientation by using an affine transformation matrix.

In order to guarantee a standard quality of *datum* for any generated virtual environment, while at the same time to provide the maximum web dissemination of the input process, only a small subset of `SVG` primitives is recognized. The parsing process will understand the *rect*, *line*, *path*, *polygon* and *polyline* elements, as well as any of their hierarchical aggregation into the container element `<g>`. Transformations applied to the `<g>` element are performed on all of its child elements. Attributes applied are inherited by child elements. In addition, it can be used to define complex objects that can later be referenced by name.

[‡] Performed on a real business tower in Rome.

© The Eurographics Association 2015.

From vector drawings to cellular complex

Line intersection The first stage concerns the computation of the intersection points among a set of line segment in the 2D plane. The containment boxes of the line segments are iteratively classified into nd-tree-like buckets and compared against the 0-dimensional centroid of smaller and smaller buckets of data. At the end of the classification, with same geometric object possibly inserted in several different buckets, a *brute-force* pairwise intersection is applied to each final subset. Finally, duplicated intersection points are removed by sorting a fixed-format coded representation, and a 1-dimensional LAR data structure is generated, with 1-cells given by the split line segments.

Maximal biconnected subgraph It is worth noting that a one-dimensional LAR model is a pair `(V,EV)` made by an array `V` of vertex coordinates and by an array `EV` of pairs of vertex indices, i.e., it is a standard graph representation. the maximal biconnected subgraph is then computed, in order to remove all dangling edges and paths, possibly generated by numerical inaccuracies and user input errors, and to generate a plane partition into a set of quasi-disjoint 2D regions. A python implementation of the standard Hopcroft-Tarjan algorithm for biconnected graph components [HT73] is used for this purpose.

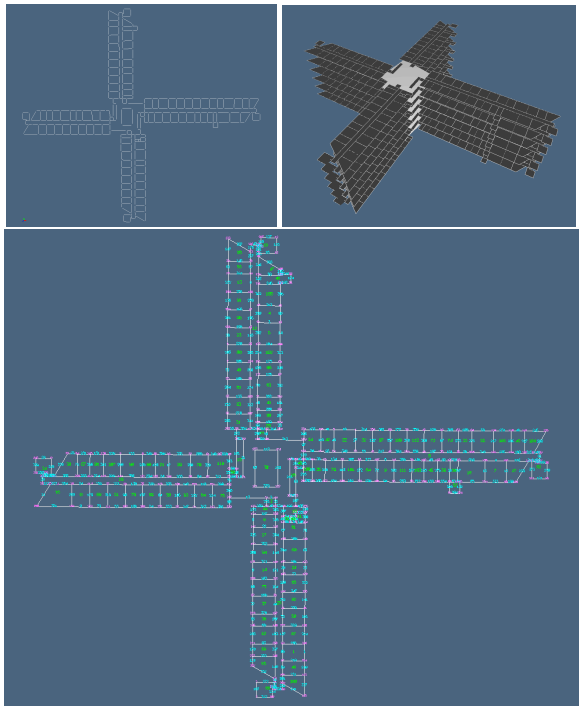


Figure 7: Exploded images of hierarchical structures: (a) 1-skeleton of the LAR model floorplan; (b) 2.5D assembly of floorplans. (c) The integer indexing `V,E,F` (a.k.a. K_0, K_1, K_2) of 0-,1- and 2-cells.

LAR of cellular complex A complete LAR of the plane partition generated by the arrangement of lines is then computed by traversing in counter-clockwise order the generated subgraphs, to report the 2-dimensional cells of the plane partition. In particular, let us consider the binary characteristic matrix `EV` as the storage for the incidence angles described below, and remark that we represent the edges on the rows, and the vertices on the columns of the matrix, so that `EV[h]=[k1,k2]` implies $e_h = (v_{k_1}, v_{k_2})$ and `EV[h,ki]`, $i \in \{1,2\}$, is the value stored in $\texttt{COO}(h,k_i,\alpha)$ when representing the sparse matrix `EV` in *coordinate format*. Hence, we can readily reconstruct the topology of 2-cells by associating to each non-zero (sparse matrix) element the angle in radians that the edge e_h forms with the horizontal line, when this one incides on the vertex v_k. Of course, if $e_h = (v_{k_1}, v_{k_2})$, then it will be

$$\texttt{EV}(h,k_2) = \texttt{EV}(h,k_1) + \pi = -\texttt{EV}(h,k_1)$$

This signed angle information is used to traverse the `EV` graph counterclockwise, and to extract the coherently oriented cycles one by one, including the exterior cycle. This traversal provides the needed basis for 2-cells, i.e. the `FV` matrix. The knowledge of both `EV` and `FV` allows to compute the ∂_2 matrix, and hence to get a complete control of the floorplan topology.

Partial ordering of hierarchical chains

Hierarchy of chains The next computational stage assigns a semantics to 2-cells and to some significant subsets of 1-cells. For this purpose we employ the concept of *hierarchy*. In mathematics, a hierarchy is a set-theoretical concept, defined as a *preorder* defined on a set. A hierarchy is often defined as either a partially or totally ordered set. All partial orders and equivalence relations are preorders, but the converse is not true. In computer science, the term usually refers to *partially ordered sets*, whose elements are classes of objects of growing complexity. In our case the preorder that defines the space hierarchy is the *relation of containment* between classes of objects. In this sense we speak of *hierarchical chains* as a family of chains (subsets of cells) partially ordered by containment.

Semantics of chains The hierarchy of building units, as follows from the functional meaning of these, is illustrated by colors in Figure 8. We may distinguish in the model floorplan of our model building five 2-chains, orderly corresponding to *north wing*, *south wing*, *east wing*, *west wing*, and *floor lobby* of the building. The four wings are further subdivided into six 2-chains: *rightside*, *leftside*, *corridor*, *elevators*, *staircases*, and *restrooms*. Each of the latter directly correspond to a subset of 2-cells. The hierarchy of spaces is described as a (nested) dictionary in `Python`, and is used to compute a unique id for each and every *space unit* and *building unit*, simply by executing the concatenation of name strings on every path of the hierarchy tree, starting from the root node.

© The Eurographics Association 2015.

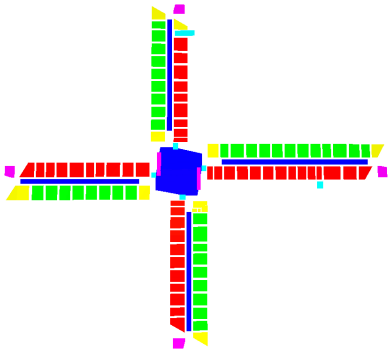


Figure 8: Hierarchical partition of 2-cells. Any hierarchical structuring is allowed. The test-case is performed on a real business tower.

Traversal of hierarchical building structure

The representation chosen for structure networks with LAR (in the `lar-cc` module) is the serialised one, consisting in ordered sequences (lists) of either (a) LAR models, or (b) affine transformations, or (c) references to other structures, either directly nested within some given structure, or called by reference (name) from within the list. In our test-case the root of the structure network is an object of `Struct` class, with name “Tower" and type “Building".

The usual aim of a structure network traversal is to transform every component structure, usually defined in a local coordinate system, into the reference frame of the structure as a whole, normally corresponding with the reference system of the structure’s root. In our case, the traversal of the *the “Tower" structure produces as output the `HIJSON` model document*, discussed in the next section.

```
def print_traversal(file, obj, parent_id='', level=0):
  if is_struct(obj):
    if is_wall(obj):
      print_wall(file, obj, parent_id)
      return
    print_struct(file, obj, parent_id)
    print_traversal(file, obj.body, obj.name)
    return
  if is_model(obj):
    print_model(file, obj, parent_id)
    return
  if is_list(obj):
    print_traversal(file, obj[0], parent_id)
    if len(obj) > 1:
      print_traversal(file, obj[1:], parent_id)
    return
  if is_mat(obj):
    print_mat(file, obj, parent_id)
    return
```

Listing 1: Algorithm producing a HIJSON file from an `obj` value.

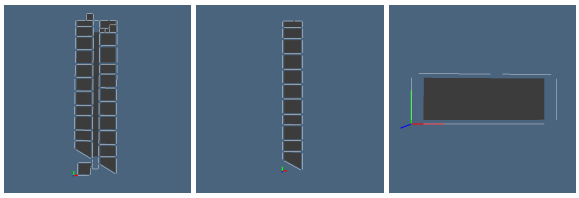


Figure 9: Sequence of local coordinates used within the hierarchical substructures of `south` structure of *floor-plan*. Let notice the position of local frames, in RGB colors.

3. HIJSON Environment

3.1. HIJSON Document

The HIJSON document is composed by a configuration section, followed by one or more FeatureCollections, containing the actual data.

The configuration part includes parameters and settings needed for building a representation in the form of a JSON Object. One of the core information in this section is defined by the correspondence between three points of the local coordinate system and three points of the real world, expressed in geographical coordinates. This is needed to ensure a seamlessly passage from local to geographical coordinate system and vice versa.

After the configuration section, the document includes a list of FeatureCollection. Each element of the list is given in the form of a GeoJSON FeatureCollection, that contains an arbitrary number of HIJSON Elements. Each FeatureCollection imposes a logical relationship that can be used to group together related HIJSON Elements. Since HIJSON Elements adhere to the GeoJSON format, each FeatureCollection results compliant with GeoJSON syntax and then accepted by any GeoJSON validator.

Dealing with indoor environments, there are essentially two classes of objects that is necessary to represent. They are (a) architectural elements, like a room, a corridor, a wall, etc. and (b) furnishings, intended in a broad sense, such as to contain both furniture, like a desk or a chair, and/or “smart objects", like an IP-cam or a connected thermostat.

Three Geometry types can be used here: Point, LineString and Polygon. The choice of the Geometry type to be associated to a HIJSON Element implicitly defines the category of the element: Point is used for furnishings, LineString for walls and doors, while Polygon may describe levels and rooms. The Geometry coordinates are expressed in metres, by convention starting at the bottom-left corner of the element, whose position is used to set-up the origin of a local coordinate frame.

© The Eurographics Association 2015.

3.2. HIJSON Toolkit

The HIJSON Toolkit is a software module that implements common operations and transformations on HIJSON documents. Written in JavaScript language, this software module has been built to be deployed in the web environment. It is modular and entirely isomorphic, i.e. can run on the server as well as on every client.

Processing pipeline

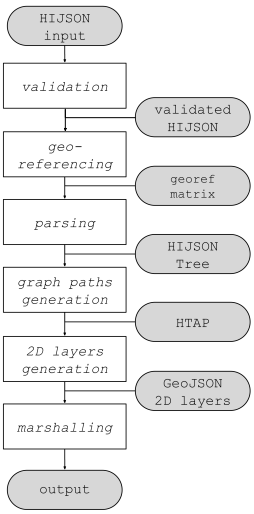


Figure 10: HIJSON Toolkit processing pipeline.

The HIJSON processing pipeline, depicted in figure 10, implements the sequence of preliminary transformations that have to be applied to a HIJSON document before any further operation. It is not strictly required to complete each stage of the pipeline: the exit stage depends on the specific use case. The application of the transformation pipeline has a double aim. The first one consists in generating the graph of valid paths among all the interesting elements. The second objective is the generation of one GeoJSON document for each storey of the building described by the HIJSON document. In this way a bidimensional layout can be provided for every level of the building, and visualized through any compliant GeoJSON viewer.

Automatic generation of valid paths

Figure 11 describes the main steps of the algorithm. The graph generated, although non optimal, ensures a complete coverage of the surface while limiting the number of generated nodes. The resulting graph is weighted on the edges with nodes distances. The graph of paths allows for calculations of directions between any two given nodes. Although different approaches have been explored [3], a very classical solution has been selected in this case, so directions are actually computed client-side by applying the Dijkstra algorithm to the graph. Taking advantage of the hierarchical structure of the HIJSON document, and according to the divide et impera approach, the problem of paths generation is split in several sub-problems, which consist in the computation of the sub- graphs relative to each individual space, more generally a single room.

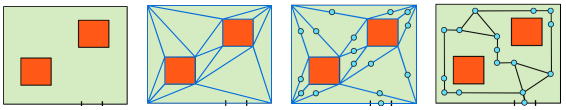


Figure 11: graph of paths generation: (a) detection of obstacles and computation of walkable area; (b) triangulation of walkable area; (c) identification of graph nodes; (d) junction of nodes.

HIJSON Class definition

Figure 12: HIJSON Element/Class/Node relationships.

To make better use of the possibilities offered by the HIJSON Toolkit and by the HIJSON document format, some custom dynamic behaviors can be described. These behaviors encapsulate the specificities relative to communication protocols with the sensors, as well as to features of user interaction. The interface for such behavior is the HIJSON Class. Every Element of the input HIJSON document has a dynamic counterpart, a running instance called Node, instantiated according to the corresponding Class via reflection methods, see figure 12.

User's needs for new indoor elements, greatly different sensor equipments, alternative representations of 2D or 3D viewports are accepted by the definition of new HIJSON Classes, that so provide single-point custom extensions of the Toolkit capabilities.

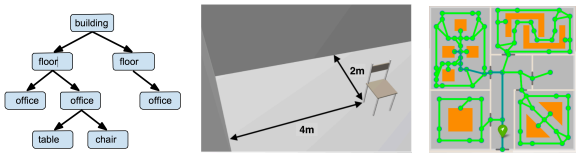
4. FIVE Web Framework

Figure 13: FIVE Web Framework UI

© The Eurographics Association 2015.

The *Framework for Indoor mapping in Virtual Environments* (FIVE), responds to the needs of an extendable, customizable, and scalable web framework which provides at the same time IoT monitoring, real-time multi-person tracking and cross-storey user navigation. Expandability and customizability derive from both design choices and HIJSON inherent characteristics, i.e. the possibility of semantic extensions. Scalability is directly borrowed from technologies used for software development: JavaScript language, using Node.js, in particular Express.js as backend framework, exploiting the power of WebSocket protocol through the Socket.io library. On the clientside it is based on libraries such as React, “the JavaScript library for building user interfaces" by Facebook, and Three.js, the current “de-facto" standard to deal with WebGL technologies.

Being supported by the web-as-a-platform, the framework exposes also an high availability: it is so simple to use as to visit a website, both from desktop or mobile devices, without explicit requirements to install any software package from proprietary stores—access to which is often denied from business devices.



(a) (b) (c)

Figure 14: HIJSON + FIVE: (a) Hierarchical structures; (b) metric positioning of furniture e/o IoT devices in the local coordinate frame; (c) automatic construction of the whole graph of paths within the building.

4.1. Applications

The Framework has been designed with focus on two different kind of users: the Explorer and the Supervisor. They have different requirements and are likely equipped with different devices: while the Supervisor monitors the indoor environment through a desktop workstation, the Explorer has a smartphone available and needs to be routed across the building. In both cases, the web platform ensures a perfect alignment with the BYOD (Bring Your Own Device) approach, nowadays often supported by companies that encourage employees to use personal devices. The numeric accuracy of our business tower virtual model gives an average error of about $10\,cm$. This one strongly depends on the resolution and size of the display of the device used for the interactive conversion of floorplans from raster images to svg vector drawigns.

4.2. IoT monitoring

An IoT monitoring application consists of an interface showing to the user, in a single, integrated and centralized way, the information collected from all the smart objects modelled in the HIJSON document. IoT monitoring application provides bidirectional communication, since the interface let the user receive information coming from smart objects while allowing him to send commands to them.

As the name itself may suggest, it is an activity specifically performed by a Supervisor user, but it can be also suitable to be deployed for the Explorer user, since she can take advantage of the interactive information coming from the surroundings objects while she moves across the indoor environment.

Monitoring different smart objects may require different ways to visualize and/or send data and commands. In particular, the user interface is characterized by a dual-display mode, that allows the user to see at the same time a 2D map that gives an overall glance in a simplified plan, and a 3D virtual environment to navigate into, as shown in Figure 15.

Alongside with typical smart objects, suitable to deal with like thermostats, where the user can read the room temperature and turn the heating on/off, other kinds of objects, that are not properly considered “smart", can be integrated into the HIJSON environment. It is the case, for example, of fire extinguishers, that are able to show the date of their last check, stored in a database.

4.3. Realtime multi-person tracking

Realtime multi-person tracking allows a Supervisor to monitor the current real position of people inside the building. This kind of task can be useful for several reasons, including security, logistics or to supervise composite operative workflows. Each device equipped with the Explorer application is in charge of locating itself, interacting with the indoor positioning system, and notifying the current position in continuous mode. Evidence of the people position is given to the Supervisor both into a 2D map and an immersive 3D virtual environment (see Figure 6).

4.4. Cross-storey user navigation

The FIVE Web Framework also provides the capability to give directions to Explorer users that must move across the indoor environment. The user specifies a starting and an ending point and the system provides him with a valid connection path. This feature strongly rely on the graph of paths generated by the Toolkit, so starting and ending points must be nodes of the graph. Connection nodes are introduced to represent stairs or elevators, enabling cross-storey paths to be computed. Since paths can span more than one storey, the most effective way to display them to the user is to show the connection nodes visualized in one or more 2D maps.

© The Eurographics Association 2015.

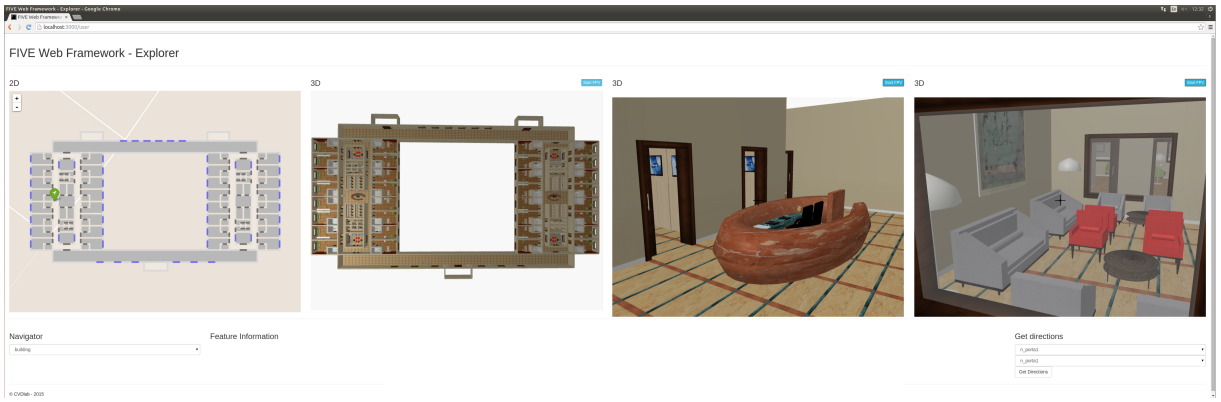


Figure 15: The modeling of two ward departments in a general hospital.

5. Conclusions

In this paper a novel document format, named HIJSON, for indoor cartographical descriptions has been introduced. Utilization of local metric coordinate frames, avoiding the manipulation of global geographical coordinates, really inconvenient when dealing with indoor spaces and objects, greatly enhances the modeling and rendering of the document content. Currently, we produce the HIJSON document from a python script using two libraries for geometric computing (pyplasm and larcc). The modeling process can be further improved by implementing a LAR-based graphical editor to assist the user during the description of the indoor space. The realization of such an editor is already in our plans.

On the basis of a HIJSON text file a web virtual environment may work as a unifying interface to run a bunch of different applications. A prototype reference architecture of such a platform has been implemented and described in this work. The architecture supports a whole range of applications: IoT monitoring, realtime multi-person tracking and user cross-storey navigation are already implemented and described. A very convenient way to extend the representation capabilities of IoT *smart objects* is also mentioned as semantic extensions. These extensions, which affects both document format and the web framework, might be easily collected in a public repository. Community could both use public available extensions or contribute by mapping new (smart) objects inside the HIJSON document format.

The main limitation of the present approach may be found in the client-side transformation of the 2.5D cellular model performed via simple polyhedral extrusions of chain descriptions made of polygons and polylines. This 3D model generation is not extendable to more unusual architectures, like, e.g. hanging structures. Anyway, even the most creative shapes could be directly modeled server-side, and inserted within HIJSON files via the same mechanism currently used for furnitures and smart-devices. We plan to inser this generalization with the publication of the HIJSON format.

Acknowledgments We would like to thank the anonimous referees for their valuable comments and suggestions. A.P. acknowledges the inspiring cooperation on LAR from Antonio DiCarlo and Vadim Shapiro. Thanks are extended to SOGEI, the ICT company of the Italian Ministry of Economy and Finance, for the support provided through several grants. Giulia Clementi and Cesare Catavitello have developed the virtual environments of the ward department and the emergency department of a general hospital model, respectively.

References

[AWW91] AIKEN A., WILLIAMS J. H., WIMMERS: The fl project: The design of a functional language, 1991. 5

[Bac78] BACKUS J.: Can programming be liberated from the von Neumann style?: a functional style and its algebra of programs. *Commun. ACM 21*, 8 (1978), 613–641. doi:http://doi.acm.org/10.1145/359576.359579. 5

[Bas10] BASAK T.: Combinatorial cell complexes and Poincaré duality. *Geometriae Dedicata 147*, 1 (2010), 357–387. URL: http://dx.doi.org/10.1007/s10711-010-9458-y, doi:10.1007/s10711-010-9458-y. 4

[BDD*08] BUTLER H., DALY M., DOYLE A., GILLIES S., SCHAUB T., SCHMIDT C.: The geojson format specification, 16 June 2008. GeoJSON is a geospatial data interchange format based on JavaScript Object Notation (JSON). URL: http://geojson.org/geojson-spec.html. 2

[BDHL*14] BOYSEN M., DE HAAS C., LU H., XIE X., PILVINYTE A.: Constructing indoor navigation systems from digital building information. In *Data Engineering (ICDE), 2014 IEEE 30th International Conference on* (March 2014), pp. 1194–1197. doi:10.1109/ICDE.2014.6816739. 2

[BG12] BULUÇ A., GILBERT J. R.: Parallel sparse matrix-matrix multiplication and indexing: Implementation and experiments. *SIAM Journal of Scientific Computing (SISC) 34*, 4 (2012), 170 – 191. URL: http://gauss.cs.ucsb.edu/~aydin/spgemm_sisc12.pdf, doi:10.1137/110848244. 4

[BWW90] BACKUS J., WILLIAMS J. H., WIMMERS E. L.: An introduction to the programming language FL. In *Research topics*

© The Eurographics Association 2015.

in functional programming. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990, pp. 219–247. 5

[CK94] CARRARA G., KALAY Y. E. (Eds.): *Knowledge-Based Computer-Aided Architectural Design*. Elsevier Science Inc., New York, NY, USA, 1994. 3

[CP80] CARRARA G., PAOLUZZI A.: *A System Approach to Building Program Planning*. Tech. Rep. CABD-Lab 80-02, Inst. of Architecture, Building and Town Planning, Sapienza University of Rome, 1980. 3

[DPS14] DICARLO A., PAOLUZZI A., SHAPIRO V.: Linear algebraic representation for topological structures. *Comput. Aided Des. 46* (Jan. 2014), 269–274. URL: http://dx.doi.org/10.1016/j.cad.2013.08.044, doi:10.1016/j.cad.2013.08.044. 3, 4

[ETSL08] EASTMAN C., TEICHOLZ P., SACKS R., LISTON K.: *BIM Handbook: A Guide to Building Information Modeling for Owners, Managers, Designers, Engineers and Contractors*. Wiley Publishing, 2008. URL: http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0470541377.html. 2

[Ger96] GERO J. (Ed.): *Advances in Formal Design Methods for CAD*. Chapman & Hall, New York, NY, USA, 1996. doi:10.1007/978-0-387-34925-1. 3

[GGM12] GOTLIB D., GNAT M., MARCINIAK J.: The research on cartographical indoor presentation and indoor route modeling for navigation applications. In *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on* (Nov 2012), pp. 1–7. doi:10.1109/IPIN.2012.6418876. 2

[Ghr14] GHRIST R.: *Elementary Applied Topology*, 1.0 ed. Createspace, 2014. URL: https://www.math.upenn.edu/~ghrist/notes.html. 4

[Hat02] HATCHER A.: *Algebraic Topology*. Cambridge University Press, 2002. URL: https://books.google.it/books?id=BjKs86kosqgC. 4

[HT73] HOPCROFT J., TARJAN R.: Algorithm 447: Efficient algorithms for graph manipulation. *Commun. ACM 16*, 6 (June 1973), 372–378. URL: http://doi.acm.org/10.1145/362248.362272, doi:10.1145/362248.362272. 7

[ind13] INDOOR.IO: Indoorjson specification and validator. https://github.com/asaarinen/indoor-json, 2013. Accessed: 2015-03-23. 2

[Pao03] PAOLUZZI A.: *Geometric Programming for Computer Aided Design*. John Wiley & Sons, Chichester, UK, 2003. URL: http://onlinelibrary.wiley.com/book/10.1002/0470013885. 5

[PDFJ15] PAOLUZZI A., DICARLO A., FURIANI F., JIRIK M.: CAD models from medical images using LAR. *Computer-Aided Design and Applications 13* (2015). To appear. 3, 5

[PMS15] PAOLUZZI A., MARINO E., SPINI F.: Lar-abc, a representation of architectural geometry from concept of spaces, to design of building fabric, to construction simulation. In *Advances in Architectural Geometry 2014*, Block P., Knippers J., Mitra N. J., Wang W., (Eds.). Springer International Publishing, 2015, pp. 353–372. URL: http://dx.doi.org/10.1007/978-3-319-11418-7_23, doi:10.1007/978-3-319-11418-7_23. 3, 4

[PPV95] PAOLUZZI A., PASCUCCI V., VICENTINO M.: Geometric programming: a programming approach to geometric design. *ACM Trans. Graph. 14*, 3 (July 1995), 266–306. URL: http://doi.acm.org/10.1145/212332.212349, doi:10.1145/212332.212349. 5

[PSB15] PENDYALA V. S., SHIM S. S., BUSSLER C.: The web that extends beyond the world. *IEEE Computer* (May 2015). 2

[Spo15] SPORTILLO M.: *FIVE - Framework per il mapping virtuale di ambienti indoor interattivi su web: Architettura di supporto allo Internet of Things*. Master's thesis, Dipartimento di Ingegneria, Università degli Studi Roma Tre, 2015. 4

[Vir15] VIRGADAMO M.: *FIVE - Framework per il mapping virtuale di ambienti indoor interattivi su web: Navigazione e Calcolo dei Percorsi*. Master's thesis, Dipartimento di Ingegneria, Università degli Studi Roma Tre, 2015. 4

[WOV*07] WILLIAMS S., OLIKER L., VUDUC R., SHALF J., YELICK K., DEMMEL J.: Optimization of sparse matrix-vector multiplication on emerging multicore platforms. In *Proceedings of the 2007 ACM/IEEE conference on Supercomputing* (New York, NY, USA, 2007), SC '07, Acm, pp. 38:1–38:12. URL: http://doi.acm.org/10.1145/1362622.1362674, doi:10.1145/1362622.1362674. 4

© The Eurographics Association 2015.