

When Pyramids Learned Walking*

Walter G. Kropatsch

PRIP, Vienna University of Technology, Austria
kpw@rip.tuwien.ac.at
<http://www.rip.tuwien.ac.at/>

Abstract. A temporal image sequence increases the dimension of the data by simply stacking images above each other. This further raises the computational complexity of the processes. The typical content of a pixel or a voxel is its grey or color value. With some processing, features and fitted model parameters are added. In a pyramid these values are repeatedly summarized in the stack of images or image descriptions with a constant factor of reduction. From this derives their efficiency of allowing $\log(\text{diameter})$ complexity for global information transmission. Content propagates bottom-up by reduction functions like inheritance or filters. Content propagates top-down by expansion functions like interpolation or projection. Moving objects occlude different parts of the image background. Computing one pyramid per frame needs lots of bottom-up computation and very complex and time consuming updating. In the new concept we propose one pyramid per object and one pyramid for the background. The connection between both is established by coordinates that are coded in the pyramidal cells much like in a Laplacian pyramid or a wavelet. We envision that this code will be stored in each cell and will be invariant to the basic movements of the object. All the information about position and orientation of the object is concentrated in the apex. New positions are calculated for the apex and can be accurately reconstructed for every cell in a top-down process. At the new pixel locations the expected content can be verified by comparing it with the actual image frame.

1 Introduction

Humans and animals are able to delineate, detect and recognize objects in complex scenes very rapidly. One of the most valuable and critical resources in human visual processing is time. Therefore a highly parallel model is the biological answer to deal satisfactorily with this resource [1]. Tsotsos [2] showed that hierarchical internal representation and hierarchical processing are the credible approach to deal with space and performance constraints, observed in human visual systems. Moreover, Tsotsos [3] concludes that in addition to spatial parallelization, a **hierarchical organization** is among the most important features of the human visual systems.

* Supported by the Austrian Science Fund under grants S9103-N13, P20134-N13 and P18716-N13.

It is now accepted that the human visual system has a hierarchical (pyramidal) architecture and that the visual mechanisms can be adequately modeled by hierarchical algorithms [4]. Pyramid algorithms are adequate models for the Gestalt rules of perceptual organization such as proximity, good continuation, etc. [5, 6]. Moreover, Privitera et al. [7] showed, in a stimulation of the human visual system, that there are two strategies to obtain and apply information about the importance of different regions of an image: the **bottom-up** methods retrieve features only from the input image, and **top-down** methods are driven by available knowledge about the world. Thus the hierarchical structure must allow the transformation of **local** information (based on sub-images) into **global** information (based on the whole image), and be able to handle both locally distributed and globally centralized information. This data structure is known as **hierarchical architecture or pyramid** [8].

The (image) pyramid might be the answer to the time and space complexity in computer vision systems, by implementing both processing strategies: bottom-up and top-down. This hierarchical structure allows distribution of the global information to be used by local processes. The main advantage of the hierarchical structures is rapid computation of a global information in a recursive manner. The change of local over to global information, e.g. from pixels arrays to descriptive data structures, is a point of discontinuity in vision systems [8]. Hierarchical structures offer a way to alleviate this discontinuity, where global structures become local in higher levels of this hierarchy.

1.1 Recall on Image Pyramids

Tanimoto [9] defines a **pyramid** as a **collection of images of a single scene at different resolutions**. In the *classical pyramid* every 2×2 block of cells is merged recursively into one cell of the lower resolution. We formally describe this structure by $2 \times 2/4$ which specifies the 2×2 reduction window and the reduction factor of 4. This type of pyramid has been extensively studied (e.g. [10], [11]).

Tanimoto's formal definitions refer to this type of pyramid [12]. He defines a cell (Tanimoto uses the term *pixel*) in a pyramid as a triple (x, y, v) which is defined in a **hierarchical domain** of n levels:

$$\{(x, y, v) | 0 \leq x \leq 2^v, 0 \leq y \leq 2^v, 0 \leq v \leq n - 1\} \quad (1)$$

Then a pyramid is any function whose domain is a hierarchical domain. This function assigns to every cell in the simplest case a value, but also structures of higher complexity can be stored.

1.2 The Flow of Information within a Pyramid

Information necessary to connect the observed part of the object with the parts in the adjacent cells must be passed up to the next lower resolution level (or equivalently, to the next higher pyramid level). There, the cells cover a larger

area and can join some parts of the level below. This process is repeated up to successively lower resolutions until the whole object is within the observation window of a cell. Unfortunately, in some pyramid structures a small rigid motion (shift, rotation) of the object may cause a completely different representation (the representation cell may be many levels below or above. [13]). This problem is resolved by the adaptive pyramid [14] which is the direct precursor of the irregular pyramid (see section 3).

An important class of operations is responsible for the bottom-up information flow within the pyramid: the **reduction function** R . It computes the new value of a cell exclusively from the contents of its children. Given an image in the base of the pyramid, application of a reduction function (e.g. average) to all first level cells fills this level. Once the cell of the first level received a value, the same process can be repeated to fill the second level and so on to the top cell. With these operations the levels $G_i, i = 0, \dots, n$ of a (Gaussian) pyramid are generated by following iterative process: $G_0 := I; G_{i+1} := R(G_i), i = 0, \dots, n - 1$

1.3 Laplacian Image Pyramids

Burt [15] describes a method for compressing, storing and transmitting images in a computationally efficient way.

Let G_k denote a $5 \times 5/4$ Gaussian pyramid, where k denotes the different levels and G_0 is the base. The bottom-up building process is based on the reduction function R : $G_k := R(G_{k-1}), k := 1, 2, \dots, n$. The reduction function maps the children's collective content into the properties of the parent.

The Gaussian smoothing filter has a low-pass characteristic removing only the highest frequencies. Therefore the Gaussian pyramid $G_k; k = 0, \dots, n$ contains a high amount of redundancy which is substantially reduced in the Laplacian pyramid:

1. The expansion function E is the reverse function of the reduction function R . It expands (interpolates) the properties of the parent(s) cells into the children's content at the higher resolution level.
2. The 'reduce - expand' RE Laplacian pyramid compares the child's content with the expanded content of the parents and simply stores the difference:

$$L_l := G_l - E(G_{l+1}) \text{ for } l := 0, 1, \dots, n - 1 \quad (2)$$

3. Reconstruction of G_k is exact: $G_k := L_k + E(G_{k+1})$ for $k := n - 1, n - 2, \dots, 0$.
4. Hence storing $G_n, L_{n-1}, L_{n-2}, \dots, L_0$ is sufficient for exact reconstruction of the original image G_0 .

Note that the intensity of the reconstructed image depends on the intensity of the apex. *If the grey value of the apex is increased the intensity of the whole reconstructed image is increased by the same value.* We observe that all the levels below the apex of the Laplacian pyramid *are invariant to global changes in intensity*.

1.4 First Steps in a Dynamic World

In [16], the Laplacian pyramid has been used to indicate a significant change in a time-series of images. Let $I(t)$ denote the image taken at time t , let m denote the level at which the change shall occur. Following procedure initiates an **alarm** when an unusual situation occurs in the field of view:

1. $D(t) := I(t) - I(t-1)$;
2. build Laplacian pyramid $L_i(t), i := 1, 2, \dots, m$ with $L_0(t) := D(t)$;
3. square level m : $L_m(t)^2$;
4. build Gaussian pyramid $G_k(t), k := 1, 2, \dots, n$ with $G_0(t) := L_m(t)^2$;
5. threshold $G_k(t), k := 1, 2, \dots, n$: alarm.

In this case the base of the Laplacian pyramid are the frame differences. It is computed bottom-up up to level m which identifies the frequency band at which the event causes the alarm. This nicely eliminates high frequency components and false-alarms caused by noise or tree branches moving in the wind.

Although this early use of pyramids for detecting dynamic changes in an image sequence was used in several applications it focused on a single event and could not filter out a description of the alarm causing event.

1.5 Some Words on Graphs

Graph hierarchies allow to use other spatial orderings of image primitives, not only the regular spatial structures like arrays. Image primitives (e.g. pixels, edges, etc.) are represented by vertices and their relations by edges of the graph. These vertices and edges are attributed. A classical example of graph representation of a set of primitives is the **region adjacency graph** (RAG), where each image region is represented by a vertex, and adjacent regions are connected by an edge. Attributes of vertices can be region area, average gray value, region statistics etc.; and attributes of edges can be the length of the boundary, the curvature, etc. between the pair of adjacent regions. The graph hierarchy is then built by aggregating these primitives. The main application area of the region based representation is *image segmentation* and *object recognition* [17]. Note that region adjacency graph (RAG) representation is capable to encode only the neighborhood relations.

1.6 and some Words on Image Segmentation

An image segmentation partitions the image plane into segments that satisfy certain homogeneity criteria (see [18] for an overview). There are many reasons for using the hierarchical paradigm in image partitioning [19]:

- the scale at which interesting structure is important is not known in advance, therefore a **hierarchical image representation** is needed;
- **efficiency of computation**: the results obtained from the coarse representation are used to constrain the costly computation in finer representations; and

- **bridging the gap** between elementary descriptive elements (e.g. pixels) and more global descriptive elements, e.g. regions (see [20]).

Although the goal of image segmentation is producing a single partition of the image, and not necessarily a hierarchy, the hierarchical representation is needed, especially if the image context is not taken into consideration. The idea behind this is if you do not know what you are looking for in an image, then use a hierarchical representation of the image, and moreover a data structure that allows the ability to access the finest partitioning (in our case the bottom of the pyramid) or in case of ‘bad’ partitioning the faculty to repair these ‘errors’. A wide range of computational vision tasks could make use of segmented images, just to mention some: object recognition, image indexing, video representation by regions etc., where such a segmentation relies on efficient computation.

1.7 Overview of the Paper

After discussing current representations of objects with both spatial and temporal structure (like articulation), we recall the basic concept of irregular graph pyramids in Section 3. Their basic properties are then efficiently applied in the new concept for describing the temporal evolution of a tracked object (Section 4). It relates the principles of the Laplacian pyramid with the graph pyramid to separate two types of information: the trajectory and the dynamic orientation is concentrated in the apex of the object (*only one ‘foot’ is updated at each step*), while all the lower levels code the spatial structure of the object if it is rigid (Section 5). Extensions lossless rotation, articulated parts and adaptive zoom are shortly addressed in Section 6. The conclusion (Section 7) summarizes the major advantages of the new proposal and lists some of the many future applications of the concept.

2 Objects with Structure in Space and in Time

In physics, motion means a change in the location of a “physical body” or parts of it. Frequently the motion of a (mathematical/geometrical) point is used to represent the motion of the whole body. However in certain cases (e.g. parking a car) more information than a single point is required. Because describing an object by an un-ordered set of all its points and their motion is not optimal (considering for example storage space, redundancy, and robustness with respect to missing or incorrect information), we can use the part structure of natural physical bodies (e.g. “objects”) to represent them in a more efficient way.

In the context of computer vision, a representation for an object can be used to model knowledge (e.g. appearance, structure, geometry) about the object and its relation to the environment. This knowledge can be used for tasks like: verifying if a certain part of an image is the object of interest, identifying invalid configurations, guiding the search algorithm for a solution/goal, etc. These tasks are in turn used by processes like segmentation, tracking, detection, recognition, etc.

Considering representations for structured objects, we identify the following spatial and temporal scales. On the **spatial scale** there are representations considering:

- i. no spatial decomposition information;
- ii. statistical information about the parts (e.g. number of parts/features of different type);
- iii. “static” structure i.e. adjacency of parts;
- iv. degrees of freedom (e.g. articulation points);
- v. pose relations between parts – correct/incorrect configurations/poses.

On the **temporal scale** we have:

- a. no temporal information;
- b. instant motion (e.g. speed and direction at a certain time instance);
- c. elementary movement (e.g. moving the arm down);
- d. action (e.g. a step, serving in tennis);
- e. activity (e.g. running, walking, sleeping, playing tennis).

On the spatial scale, representations cover the whole domain from i. to v. (see [21–24]). There are simple representations like *points* [25–27], *geometric shapes* (rectangle, ellipse) [28, 29], and *contours/silhouettes* [30, 31], but also more complex ones [32, 33]. Felzenszwalb et al. [32] use pictorial structures to estimate 2D body part configurations from image sequences. Navaratnam et al. [33] combine a hierarchical kinematic model with a bottom up part detection to recover the 3D upper-body pose. In [34] a model of a hand with all degrees of freedom and possible poses is used.

On the temporal domain, most methods use simple motion models, typically considering the motion between a few consecutive frames. More complex representation on the temporal domain can be found in *behavior understanding*, where dynamic time warping (e.g. [35]), finite-state machines (e.g. [36]), and hidden Markov models (e.g. [37, 38]) are employed. In the fields of pose estimation and action recognition there is a so-called state space representation. For example a human can be represented by a number of sticks connected by joints [39]. Every degree of freedom of this model is represented by an axis in the state space. One pose of a human body is one point in this high-dimensional space and an event/action is a trajectory in this space. This trajectory through the state space is one possibility to represent the temporal aspect [40]. Nevertheless, there are still very few works that look at complex spatial and temporal structure at the same time (e.g topology in the 4D spatio-temporal domain).

In the context of computer vision, properties relating the objects with the visual input also need to be represented. Considering the **dynamics of a description** created using a certain representation, one can look at how *a description* and its *building/adapting processes* behave, when the represented information changes. For example: number of parts or their type, static structure, type of activity, relation to visual input (scaling, orientation), etc.

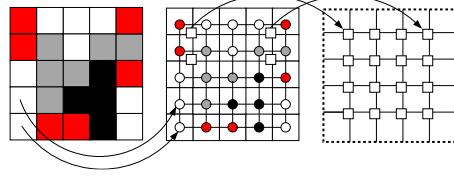


Fig. 1. Image to primal and dual graphs.

For small changes in the information a minimal change in the description is desired. E.g. scaling, rotation, part articulation, illumination, should only minimally affect the description.

In addition to the dynamics, one can talk about the **genericness** of a representation i.e. the ability to represent objects of varying degree of complexity and abstraction (e.g. industrial robot, normal human walking, stone).

3 Irregular Graph Pyramids

Pyramids can be built also on graphs. In this case the domain is no more the simple array structure as in Tanimoto's definition but a graph where the function values are stored as attributes of the vertices of the graph. A RAG encodes the adjacency of regions in a partition. In the simplest case a vertex corresponds to a pixel and the edges encode the 4-neighborhood relations (Fig. 1). The dual vertices correspond in this case to the centers of all 2×2 blocks, the dual edges are the cracks between adjacent pixels. More generally, a vertex can be associated to a region, vertices of neighboring regions are connected by an edge. Classical RAGs do not contain any self-loops nor parallel edges. An *extended region adjacency graph*



Fig. 2. A digital image I , and boundary graphs \bar{G}_6 , \bar{G}_{10} and \bar{G}_{16} of the pyramid of I .

(eRAG) is a RAG that contains some *pseudo edges*. Pseudo edges are the self-loops and parallel edges that are required to encode neighborhood relations to a cell *completely enclosed* by one or more other cells [41] i.e. they are required to correctly encode the topology. The *dual graph* of an eRAG G is called the *boundary graph* (BG, see Fig. 2) and is denoted by \bar{G} . The edges of \bar{G} represent the boundaries (borders) of the regions encoded by G , and the vertices of \bar{G} represent points where boundary segments meet. G and \bar{G} are planar graphs. There is a one-to-one correspondence between the edges of G and the

edges of \bar{G} , which also induces a one-to-one correspondence between the vertices of G and the 2D cells (will be denoted by *faces*¹) of \bar{G} . The dual of \bar{G} is again G . The following operations are equivalent: edge contraction in G with edge removal in \bar{G} , and edge removal in G with edge contraction in \bar{G} .

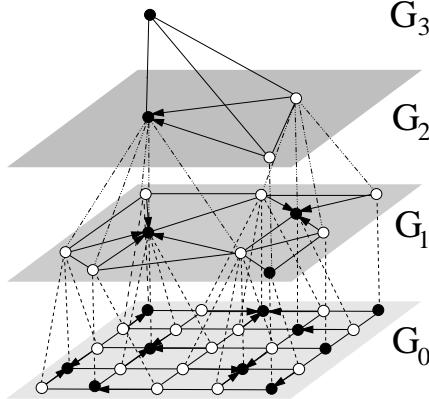


Fig. 3. Example graph pyramid.

A (dual) irregular graph pyramid [41, 42] is a stack of successively reduced planar graphs $P = \{(G_0, \bar{G}_0), \dots, (G_n, \bar{G}_n)\}$ (Fig. 3). Each level (G_k, \bar{G}_k) , $0 < k \leq n$ is obtained by first contracting edges in G_{k-1} (removal in \bar{G}_{k-1}), if their end vertices have the same label (regions should be merged), and then removing edges in G_{k-1} (contraction in \bar{G}_{k-1}) to simplify the structure. The contracted and removed edges are said to be *contracted* or *removed* in (G_{k-1}, \bar{G}_{k-1}) . In each G_{k-1} and \bar{G}_{k-1} the contracted edges form trees called *contraction kernels*. One vertex of each contraction kernel is called a *surviving vertex* and is considered to have ‘survived’ to (G_k, \bar{G}_k) . The vertices of a contraction kernel in level $k-1$ form the *reduction window* of the respective surviving vertex v in level k . The *receptive field* of v is the (connected) set of vertices from level 0 that have been ‘merged’ to v over levels $0 \dots k$.

4 Moving Objects

The study of dynamic image sequences (or videos) aims at identifying objects in the observed image sequence and describing their integrated properties and their dynamic behaviour. There are several possibilities to segment an object from an image or a video:

¹ Not to be confused with the vertices of the dual of a RAG (sometimes also denoted by the term *faces*).



Fig. 4. Example of an extracted object and its rigid parts.

- image segmentation methods are able to locate image regions in individual images that are ‘homogeneous’ in certain terms. Examples are David Lowe’s SIFT-features [43], different variants of Ncut [44] or the MST pyramid [45]. Objects of interest are, however, mostly composed of several such regions and further grouping is required.
- Optical flow approaches overcome the grouping since the different parts of an object usually move together.
- detection of interest points and tracking them individually over the sequence. In order to preserve the structure of points belonging to the same object pairwise relations like distances can be used efficiently to overcome failures caused by noise or occlusions (see [46, 47]).

4.1 Extraction of Structure From Videos

In [47] a graph-pyramid is used to extract a moving articulated object from a video, and identify its rigid parts. First a spatio-temporal selection is performed, where the spatial relationships of tracked interest points over time are analysed and a triangulation is produced, with triangles labeled as *potentially-rigid* and *non-rigid*. The *potentially-rigid* triangles are given as input to a grouping process that creates a graph pyramid such that the each top level vertex represents a rigid part in the scene. The orientation variation of the input triangles controls the construction process and is used to compute the similarity between two regions. This concept is related to the single image segmentation problem [17], where the results should be regions with homogeneous color/textured (small internal contrast) neighbored to regions that look very different (high external contrast). In our case the “contrast” is interpreted as the inverse of “rigidity”. The result of this method can be used to initialize an articulated object tracker. Fig. 4 shows an example.

4.2 Describing the Tracking Results

Most of the current approaches describe the results in the domain of the original data and use the image and frame coordinates. The resulting trajectory consists in a sequence of frame coordinates where the object was at the respective time instance. We consider the use of a separate data structure for each moving object in order to update independent movements and properties in clearly separated data (i.e. to describe 'walking').

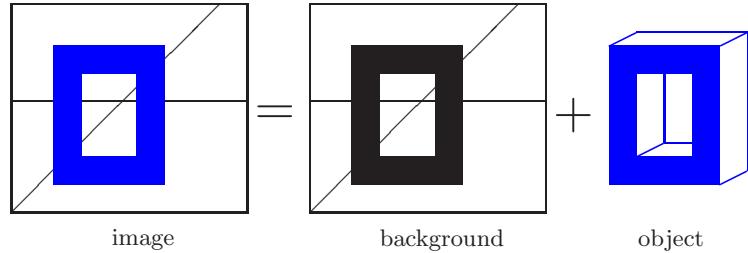


Fig. 5. Separating the object from the background

Once an object is identified in an image (frame) or even in an image pyramid we cut out the object from its pixel based representation into the neighborhood graph of pixels, close its surface topologically by invisible surface patches of the backside (Fig. 5). The remaining image is considered as background and the pixels of the removed object are labelled as invisible.

4.3 Topological Completion: Represent a 3D Object

In a video frame, a 3D object may be occluded or partially visible. We call the visible part of the surface the *front surface*. From a single image frame, the front surface is extracted as a graph. This extracted graph embeds the topological structure and discriminative visible features of the object. In the vertex, attributes like size, color and position of the corresponding pixels (region) can be stored and the edges specify the spatial relationships (adjacency, border) between the vertices (regions)[46]. Topological completion closes the visible surface by one or more invisible surface patches in order to completely cover the surface of the volumetric object.

Each level of the irregular graph pyramid is a graph, presenting the closed surface of the moving object in multiple resolutions. We collect the topological structures from the visible surface of the target object. Each graph embeds both features and structural information. Locally, features describe the object details; globally, the relations between features encode the object structure.

For initialization, the base graph of the pyramid encodes the initial information about the object, the graph is closed on the invisible backside to create a

closed 2D manifold. The graph pyramid can cope with this structure and the same operations can be applied as in the case of an image. As new visible parts of the surface would reveal previously invisible parts, the object representation is incrementally updated automatically from observing the target object in a video sequence. This requires the registration of the visible parts and the replacement of some invisible patches. When some hidden structure appears, we add the new topological structure into the previous 2D manifold to obtain the updated object representation. For instance, a rotating cup will reveal the handle that was hidden before, and hide the logo when it moves out of sight.

When the camera has covered all the aspects of the object, which means all the observable parts of the object have been integrated in the object model, the topological structure of the target object is complete. This is the process we defined as *topological completion*.

5 Walking: Only one Foot Leaves Contact with the Ground

In the image frame every pixel establishes a contact between the moving object and the digital image. In order to reduce efforts of updating large amounts of data (e.g. geometrically transforming the object window) we reduce the contact to a single point which serves as a reference similar to *the foot making the next step in walking*.

5.1 Invariance to Translation

In order to keep the geometric information of the object's surface patches we attribute each cell $v \in V$ with the coordinates of the corresponding image pixels, $p(v) = (x, y) \in [0, N_x] \times [0, N_y]$. These coordinates could, if necessary, be enhanced by depth values, $p(v) = (x, y, d) \in [0, N_x] \times [0, N_y] \times \mathbb{R}$, coming from different '*shape from X*' methods (e.g. [48]).

Both the extracted objects and the remaining background image can be embedded in an irregular graph pyramid either

- by using the existing image pyramid (e.g. after segmentation) or
- by rebuilding the pyramids of the objects and the background.

The coordinates of the higher level cells can be computed from the children either by inheritance from the surviving child to the parent or by a weighted average of the children's coordinates or by a combination with the selection of survivors such that the largest region survives and inherits its children's coordinates in the case the pyramid is rebuilt. After this bottom-up propagation each cell has 2D or 3D coordinates.

The resulting position attributes $p(v)$ are as redundant as the grey values of a Gaussian pyramid. Hence the idea of expanding the parent's coordinates $p(v_p)$ to the children, $p(c)$, $\text{parent}(c) = v_p$, and storing simply the difference vector $d(c) = p(c) - E(p(v_p))$ between the expansion and the original attribute

in analogy to the Laplacian pyramid². Let us call the difference $d(c)$ the child's **correction vector**. Similar to the Laplacian pyramid the original position of each cell can be reconstructed accurately (up to numerical precision) by adding all the correction vectors (following the equivalence $p(c) = E(p(v_p)) + d(c)$) up to the apex (a sort of equivalent correction vector). The position of the cell is then the position of the apex added to the sum of correction vectors

$$p(c_0) = p(\text{apex}) + \sum_{c=c_0, \text{parent}(c_0), \dots}^{\text{apex}} d(c).$$

As a side effect the object can be rigidly shifted by simply translating the apex to the desired position and reconstructing the coordinates of all the other cells if needed. This shift invariance of the lower pyramid levels allows simple modifications and efficient reconstruction but needs further adaptation in order to cope with rotation and scale changes.

5.2 Invariance to Rotation and Scale

So far the position of an object is coded in the coordinates $p(\text{apex})$ of the apex. Every cell below the apex contains correction vectors $d(c)$ allowing accurate reconstruction of its position by top-down refinement using the correction vectors.

Most objects have an orientation $o \in \mathbb{R}^3$ in addition to their position $p \in \mathbb{R}^3$. Orientation can be derived from properties like symmetry, moving direction or can be given by the object model a priori. Since orientation is a global property of an object we add it to the properties of the apex of the object's pyramid. The vector $o(\text{apex})$ codes both the orientation with respect to the reference coordinate system and a scale if the length $\|o\| \neq 1$ is different from unit length. Orientation and position allow to quickly transform the object pyramid from one coordinate system to another (i.e. of another camera or viewpoint).

The orientation of the object can be used to make correction vectors invariant to rotation and scale. Taking $p(v_p)$ as the position where both the orientation vector and the correction vector start we can express the correction vector $d(c)$ as a rotated and scaled version of the orientation: $d(c) = \lambda R_x(\alpha)R_y(\beta)R_z(\gamma)o$ and store the parameters $r(c) = (\lambda, \alpha, \beta, \gamma)$ as new parameters of the cell c . The angles α, β, γ can be the Euler angles of the corresponding rigid body and the scale factor $\lambda = \|d(c)\|/\|o\|$ relates the vectors' lengths. Given the position $p(v_p)$ of the parent and the orientation o of the object each cell can accurately reconstruct its position $p(c) = p(v_p) + \lambda R_x(\alpha)R_y(\beta)R_z(\gamma)o$. We note that in addition to the invariance with respect to translation, the parameters $r(c)$ are invariant also to rotation and scale. The rotation of the object is executed by applying the rotation to the orientation of the apex and similar with a scale change.

All the vertices can be accessed efficiently from the apex by following the parent - children path. The construction of the pyramid proceeds bottom - up while the reconstruction from the apex is a top - down process. In such way we can reconstruct the whole pyramid by only locating the apex point.

² In the simplest case, expand by projection, $E(x) = x$.

6 A Sequence of ‘Steps’

Now when analyzing a video sequence it is not necessary to compute one pyramid for each frame, it is enough to apply all the transformations to the apex and only to reconstruct the whole structure at the end of the process. Or we can rely to a few distinctive interest points (as done by several other approaches) the position and characteristics of which are known within the new object pyramid together with their mutual spatial relationships and track them while enforcing the preservation of the spatial relations like in the spring-system approach.

In that way, the complexity and the computation time are reduced what allows to adapt to the changes in the image frame in a more efficient way and being fast enough to deal with real time processing requirements.

6.1 Lossless Rotation

Another significant advantage of the above object pyramid is that the connectivity of both the foreground and the background is always preserved. This is not always true for other image processing tools (e.g. Photoshop), for example, when working with thin and elongated objects. Fig. 6 shows an example of a thin line (Fig. 6 a)) which is rotated by 50 degrees. As the new coordinates of the points of the line do not correspond to integer coordinates most of the image processing tools interpolate and resample the rotated coordinates in order to obtain the new position of the points. This results in a disconnected line (Fig. 6 b)) or in a thicker line if bilinear interpolation or anti-aliasing is applied. In the images of Fig. 6 b), the (red) stars mark the new rounded coordinates of each point and the black squares show the position of the points estimated by the processing tool. In our approach each pyramid level is a graph and the relations between adjacent regions are defined by the edges of the graph. When the top-down reconstruction is done, the position of each cell in each level is updated according to its correction vector but the edges of the graphs are always connecting the same vertices independently of their position. Therefore the region adjacency relations and connectivity are preserved (Fig. 6 c)).

6.2 Articulated Objects

Our approach can be extended to articulated objects. Articulated objects are characterized by two or more rigid parts joint by articulation points. The nodes of the graph corresponding to the articulation point as well as the ones corresponding to the rigid parts are identified in the graph pyramid [47]. The nodes that compose each of the rigid parts will be merged in one apex in a certain pyramid level so that to follow the movement of each rigid part it is only needed to apply the geometric transformations in its apex and then doing the top-down reconstruction.

The top row of the Fig. 7 shows the movement of an arm in a sequence of 5 video frames. For the process of tracking the movement of the arm, in the first

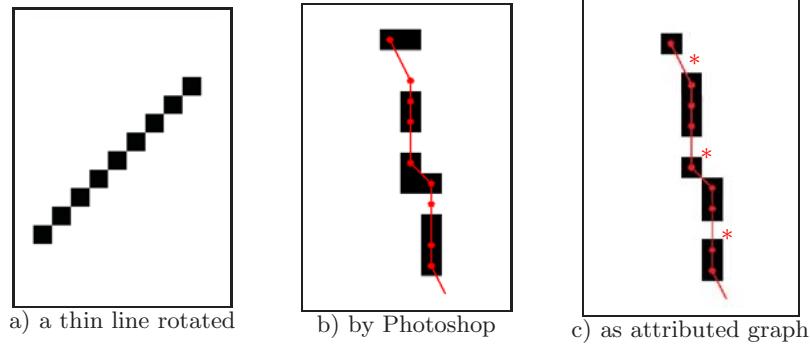


Fig. 6. 50° rotation of a thin line.

frame the structure is initialized and the pyramid is built (Fig. 7 a)). In this case only the lower part of the arm is moving, so that it will be only needed to apply all the transformations in the apex of the set of nodes that correspond to this part of the arm and reconstruct the graph at the end (7 b)). All the other nodes in the structure will remain in the same position. In that way, the tracking of articulated objects can be facilitated.

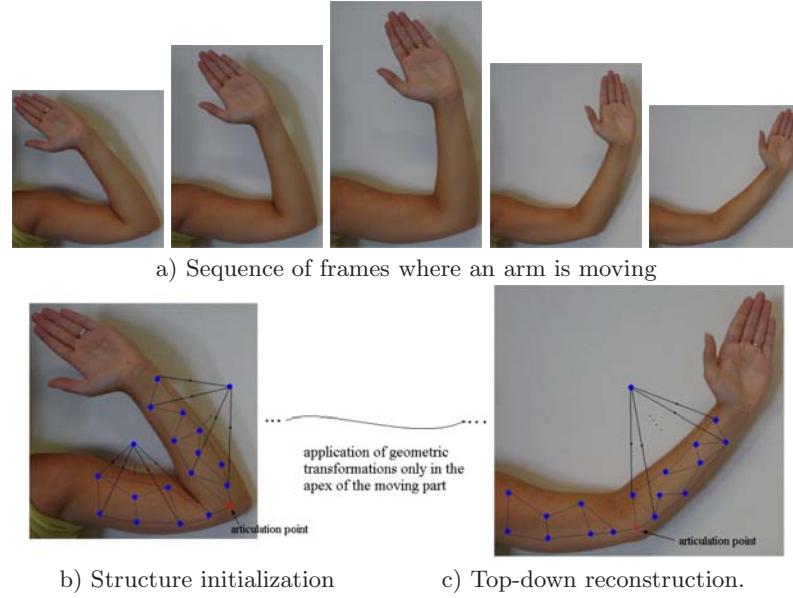


Fig. 7. Example with an articulated object.

6.3 Adaptive Zoom-In: an Approaching Object

One application of the new object pyramid is the incremental update of the object description when the object approaches the camera. If the object gets closer to the camera, the distance camera-object decreases while the resolution of the object increases. We obtain a bigger picture of the object with more details to be inserted into the existing model of the object.

From the irregular graph pyramid perspective, this new image can be seen as a projection of the original base graph which includes more details. The pyramid will expand one level below the current base graph, this new base graph encodes both structures due to a higher resolution of the object. The new base graph is relinked to the current pyramid by finding the vertical connection between the new base graph and the existing upper part of the pyramid.

Assumption. We assume the approaching speed is not too fast. The current size of the target object cannot exceed twice as the one in previous image frame. This means the maximum scaling factor cannot exceed 2. Otherwise there might be a gap between the new base graph and the previous base graph so that we have to insert extra levels to bridge the new base graph with the old base graph.

The integration of several resolutions is also needed in surveillance applications involving multiple cameras observing the same area. The observed object will be closer to some cameras but further away from others. This creates the need to integrate the different views into a consistent description.

7 Conclusion

This paper presented some aspects of the development of image pyramids from stacks of arrays to a stack of graphs describing object's surfaces at multiple resolutions and multiple levels of abstraction. By decoupling the object's description from the projected view in an image frame into an object centered pyramid representation several operations become feasible: moving the object modifies only one cell of the structure much like the step of the foot when walking: the apex.

For a rigid object, the structure of the object pyramid is invariant to basic geometric transformation, such as translation, scaling and rotation. All the information about position and orientation of the object is concentrated in the apex. All the lower levels of a rigid object are invariant to translation, rotation and scale changes but still allowing accurate reconstruction of the object's geometry.

Tracking of structured objects is facilitated by the fact that the pyramid relates the different tracked points and can compensate tracking failures in case of partial occlusions. Articulation between rigid parts can be expressed by first selecting one of the related parts as the parent and describing the articulation by the change in Euler angles in the apex of the child. Different moving object pyramids can be related by superimposing a graph describing their spatial arrangement. In this graph the apexes of the objects appear as nodes related by edges describing the particular neighborhood relations. In some cases this graph could be embedded in \mathbb{R}^3 using a 3D combinatorial pyramid [49]. In the

future several applications of the new concept are promising besides the spatio-temporal tracking of moving objects, i.e. the integration of views from different view points for surveillance.

Acknowledgements

The presented work is the result of the collaboration within the working group on robust hierarchies of PRIP. Substantial contributions came from Nicole Artner, Adrian Ion, Esther Antunez-Ortiz, Luis Alfredo Mateos, Dan Shao, Yll Haxhimusa, Helena Molina Abril, and Mabel Iglesias Ham. All these contributions are gratefully acknowledged.

References

1. Feldman, J.A., Ballard, D.H.: Connectionist models and their properties. *Cognitive Science* (6) (1982) 205–254
2. Tsotsos, J.K.: On the relative complexity of passive vs active visual search. *Intl. J. Computer Vision* **7**(2) (1992) 127–141
3. Tsotsos, J.K.: How does human vision beat the computational complexity of visual perception? In Pylyshyn, Z., ed.: *Computational Processes in Human Vision: An Interdisciplinary Perspective*. Ablex Press, Notwood, NJ (1988) 286–338
4. Zeki, S.: *A Vision of the Brain*. Oxford: Blackwell (1993)
5. Pizlo, Z., Salach-Golyska, M., Rosenfeld, A.: Curve detection in a noisy image. *Vision Research* **37**(9) (1997) 1217–1241
6. Pizlo, Z.: Perception viewed as an inverse problem. *Vision Research* **41**(24) (2001) 3145–3161
7. Privitera, C.M., Stark, L.W.: Algorithms for defining visual regions-of-interest: Comparison with eye fixations. *IEEE Tr. Pattern Analysis and Machine Intelligence* **22**(9) (2000) 970–982
8. Jolion, J.M., Rosenfeld, A.: *A Pyramid Framework for Early Vision*. Kluwer (1994)
9. Tanimoto, S.L.: Paradigms for pyramid machine algorithms. In Cantoni, V., Levialdi, S., eds.: *Pyramidal Systems for Image Processing and Computer Vision*. Volume F25 of *NATO ASI Series*. Springer-Verlag Berlin, Heidelberg (1986) 173–194
10. Tanimoto, S.L., Klinger, A., eds.: *Structured Computer Vision: Machine Perception through Hierarchical Computation Structures*. Academic Press, New York (1980)
11. Rosenfeld, A., ed.: *Multiresolution Image Processing and Analysis*. Springer, Berlin (1984)
12. Tanimoto, S.L.: From pixels to predicates in pyramid machines. In Simon, J.C., ed.: *Proceedings of the COST-13 workshop 'From the Pixels to the Features'*. AFCET, Bonas, France (August 1988)
13. Bister, M., Cornelis, J., Rosenfeld, A.: A critical view of pyramid segmentation algorithms. *Pattern Recognition Letters* **Vol. 11**(No. 9) (September 1990) pp. 605–617
14. Jolion, J.M., Montanvert, A.: The adaptive pyramid, a framework for 2D image analysis. *Computer Vision, Graphics, and Image Processing: Image Understanding* **55**(3) (May 1992) pp.339–348

15. Burt, P.J., Adelson, E.H.: The Laplacian pyramid as a compact image code. *IEEE Transactions on Communications* **Vol. COM-31**(No.4) (April 1983) pp.532–540
16. Anderson, C.H., Burt, P.J., van der Wal, G.S.: Change detection and tracking using pyramid transform techniques. *Intelligent Robots and Computer Vision SPIE* **Vol.579** (Sept.16-20 1985) pp.72–78
17. Kropatsch, W.G., Haxhimusa, Y., Ion, A.: Multiresolution Image Segmentations in Graph Pyramids. In Kandel, A., Bunke, H., Last, M., eds.: *Applied Graph Theory in Computer Vision and Pattern Recognition*. Volume 52. Springer Wien New York (2007) 3–41
18. Pal, N.R., Pal, S.K.: A review on image segmentation techniques. *Pattern Recognition* **26**(3) (1993) 1277–1294
19. Nacken, P.F.: Image segmentation by connectivity preserving relinking in hierarchical graph structures. *Pattern Recognition* **28**(6) (June 1995) 907–920
20. Keselman, Y., Dickinson, S.: Generic Model Abstraction from Examples. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* **27**(7) (July 2005) 1141–1156
21. Hu, W., Tan, T., Wang, L., Maybank, S.: A survey on visual surveillance of object motion and behaviors. *IEEE Transactions on Systems, Man and Cybernetics* **34** (2004) 334–352
22. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. *ACM Comput. Surv.* **38**(4) (2006) Article 13
23. Moeslund, T.B., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding* **104** (2006) 90–126
24. Moeslund, T.B., Hilton, A., Krüger, V.: A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding* **81** (2001) 231–268
25. Veenman, C.J., Reinders, M.J.T., Backer, E.: Resolving motion correspondence for densely moving points. *IEEE Tr. Pattern Analysis and Machine Intelligence* **23**(1) (January 2001) 54–72
26. Serby, D., Koller-Meier, S., Gool, L.V.: Probabilistic object tracking using multiple features. In: *International Conference of Pattern Recognition*, IEEE Computer Society (2004) 184–187
27. Shafique, K., Shah, M.: A non-iterative greedy algorithm for multi-frame point correspondence. In: *ICCV*. Volume 1., Nice, France, IEEE Computer Society (2003) 110–115
28. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. *IEEE Tr. Pattern Analysis and Machine Intelligence* **25**(5) (2003) 564–575
29. Csaba Beleznai, B.F., Bischof, H.: Human detection in groups using a fast mean shift procedure. In: *International Conference on Image Processing*, IEEE Computer Society (2004) 349–352
30. Yilmaz, A., Xin, L., Shah, M.: Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Tr. Pattern Analysis and Machine Intelligence* **26**(11) (November 2004) 1531–1536
31. Yunqiang, C., Yong, R., Huang, T.S.: JPDAF based HMM for real-time contour tracking. In: *CVPR*. Volume 1., Kauai, HI, USA, IEEE Computer Society (December 2001) 543–550
32. Felzenszwalb, P.F.: Pictorial structures for object recognition. *International Journal for Computer Vision* **61** (2005) 55–79
33. Navaratnam, R., Thayananthan, A., Torr, P.H.S., Cipolla, R.: Hierarchical part-based human body pose estimation. In: *British Machine Vision Conference*. (2005)

34. Bray, M., Koller-Meier, E., Schraudolph, N.N., Gool, L.J.V.: Fast stochastic optimization for articulated structure tracking. *Image Vision Comput.* **25**(3) (2007) 352–364
35. Bobick, A.F., Wilson, A.D.: A state-based approach to the representation and recognition of gesture. *IEEE Tr. Pattern Analysis and Machine Intelligence* **19** (1997) 1325–1337
36. Wilson, A.D., Bobick, A.F., Cassell, J.: Temporal classification of natural gesture and application to video coding. In: Conference on Computer Vision and Pattern Recognition. (1997) 948–954
37. Starner, T., Weaver, J., Pentland, A.: Real-time american sign language recognition using desk and wearable computer based video. *IEEE Tr. Pattern Analysis and Machine Intelligence* **20**(12) (1998) 1371–1375
38. Oliver, N., Rosario, B., Pentland, A.: A bayesian computer vision system for modeling human interactions. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(8) (2000) 831–843
39. Wren, C.R., Pentland, A.P.: Dynamic modeling of human motion. In: International Conference on Automatic Face and Gesture Recognition, IEEE Computer Society (1998) 14–16
40. Pavlovic, V., Rehg, J.M., Cham, T.J., Murphy, K.P.: A dynamic bayesian network approach to figure tracking using learned dynamic models. In: International Conference on Computer Vision. Volume 1., IEEE Computer Society (1999) 94–101
41. Kropatsch, W.G.: Building irregular pyramids by dual graph contraction. *Vision, Image and Signal Processing* **142**(6) (December 1995) 366–374
42. Kropatsch, W.G., Haxhimusa, Y., Pizlo, Z., Langs, G.: Vision Pyramids that do not Grow too High. *Pattern Recognition Letters* **26**(3) (2005) 319–337
43. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* **60**(2) (2004) 91–110
44. Shi, J., Malik, J.: Normalized cuts and image segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, IEEE (1997) 731–737
45. Haxhimusa, Y., Kropatsch, W.G., Pizlo, Z., Ion, A.: Approximative Graph Pyramid Solution of the E-TSP. *Image and Vision Computing* **27**(7) (2009) 887–896
46. Artner, N.M., López Márquez, S.B., Beleznai, C., Kropatsch, W.G.: Kernel-based Tracking Using Spatial Structure. In Arjan Kuipers, Bettina Heise, L.M., ed.: Challenges in the Biosciences: Image Analysis and Pattern Recognition Aspects. Proceedings of 32nd OEAGM Workshop, Linz, Austria, OCG-Schriftenreihe books@ocg.at, Österreichische Computer Gesellschaft (2008) 103 – 114 Band 232.
47. Artner, N.M., Ion, A., Kropatsch, W.G.: Rigid Part Decomposition in a Graph Pyramid. In Eduardo Bayro-Corrochano, J.O.E., ed.: The 14th International Congress on Pattern Recognition, CIARP 2009. LNCS, Springer (September 2009)
48. Zhang, R., Tsai, P.S., Cryer, J., Shah, M.: Shape from Shading; A Survey. *IEEE Trans. on Pattern Analysis and Machine Intelligence* **21**(8) (August 1999) 609–706
49. Illetschko, T.: Minimal combinatorial maps for analyzing 3d data. In: Technical Report PRIP-TR-110,, Institute f. Automation 183/2, Dept. for Pattern Recognition and Image Processing, TU Wien, Austria (November 2006)