

A Uniform Indexing Scheme for Geo-spatial Data and Authorizations

Vijayalakshmi Atluri
MSIS Department and CIMIC
Rutgers University, Newark, NJ
atluri@cimic.rutgers.edu

Pietro Mazzoleni¹
Dipartimento di Scienza Dell'informazione
Universita' di Milano, Milano, Italy
mazzolen@dsi.unimi.it

Abstract

High-resolution satellites are promising technologies to bolster global transparency by providing unprecedented access to accurate and timely information. A significant number in this satellite fleet are owned by private organizations, which can collect images of human activities and the environment, and make them commercially available around the globe. Uncontrolled dissemination of this information may have grave implications on national security and personal privacy, as some governments and private groups may exploit this information for aggressive purposes. Formal policies for prohibiting the release of imagery beyond a certain resolution, and notifying when an image crosses an international boundary or when such a request is made, are beginning to emerge.

As such, there is a need for effective and efficient schemes for facilitating controlled dissemination of satellite imagery and the information products generated from it. Unlike conventional authorizations that can be implemented as access control lists, since authorizations on geospatial imagery involve spatial attributes, managing the authorization base and searching for authorizations based on the spatial extent require a spatial indexing structure. An access request thus requires a search on two index structures, one for the authorizations, and the other for the image database. In this paper, we propose an indexing structure, called *RMX-quadtree*, that is capable of indexing multi-resolution satellite imagery and additionally allow representing authorizations on top of it. By employing a uniform index for both image database and authorization base, access requests can be processed more efficiently as only one index structure need to be traversed. We demonstrate how authorizations with privilege modes such as *view* and *zoom-in* can be implemented using the *RMX-quadtree*.

¹ The work of Pietro Mazzoleni was conducted during his visit to Center for Information Management, Integration and Connectivity (CIMIC), Rutgers University (NJ).

1. Introduction

High resolution² satellites are promising technologies to bolster global transparency by providing unprecedented access to accurate and timely information on many important, perhaps controversial, developments. Due to the fact that a significant number in this satellite fleet are owned by private organizations, these can collect images of human activities and the environment and make them commercially available around the globe. There are now more than 15 commercial satellites with resolutions from 1 - 30 meters. For example, satellites such as IKONOS (launched in September 1999), ORBVVIEW (OrbView-3 scheduled to be launched in Q2 of 2002), EROS and QUICKBIRD are privately owned and provide images with resolution of 1 meter or smaller. One-meter imagery enables the viewing of houses, automobiles and aircrafts, and will make it possible to create highly precise digital maps and three-dimensional fly-through scenes. A portion of the high-resolution image of the downtown New York City,³ taken on January 22, 2002 is shown in Figure 1, where one can see roads, boats and cars. ORBVVIEW proposes an application (ORBVVIEW City) including an archive of high resolution images with most of the major U.S. and non-U.S. urban areas expects global near real-time one-meter imagery. Such images are available at prices as low as \$10 and one would expect improved quality at better prices in future. Usually, the command link from the satellite to the ground station is encrypted and authenticated so that no unauthorized entity is capable of downloading the imagery. Sometimes, these systems also have capability to "listen" for commands to collect imagery.

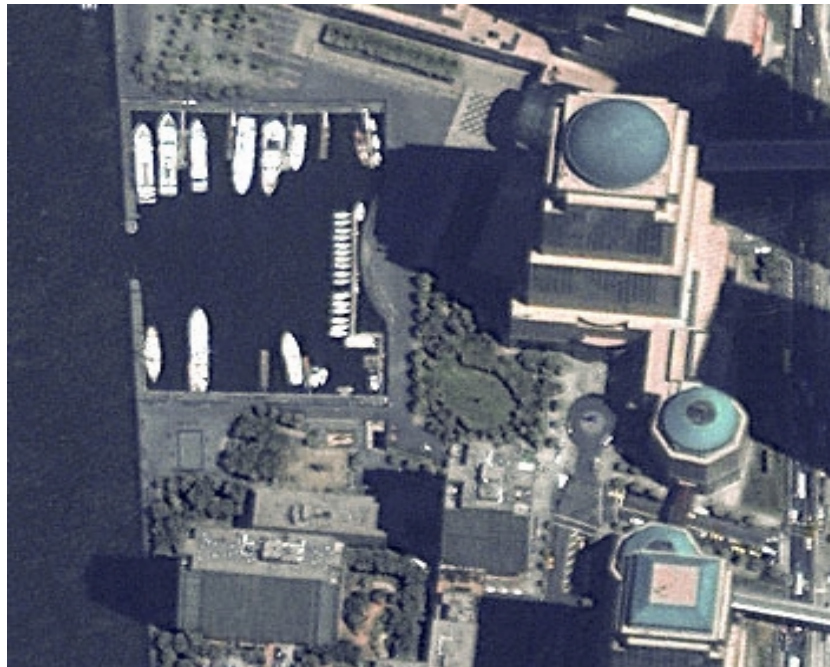


Figure 1: Downtown New York City area with 1-meter resolution

As low cost high resolution images become available, it is not unrealistic in near future to have information products such as that of Microsoft TerraServer [Barc00] in which a data warehouse of image pyramids constructed from aerial, satellite, and topographic images of the earth.⁴ With Terraserver, users can search the data warehouse by coordinates and place names, and can view the images with different

² A resolution of 5m or smaller is generally accepted as a high-resolution image.

³ A clearer image can be viewed at: www.spaceimaging.com/gallery/wtc_1_22_02.jpg.

⁴ Some preliminary products of this nature can be found at: www.spaceimaging.com

resolutions by simply clicking on them. With such data warehouses built with images taken at closer time intervals, one can attain near real-time surveillance of a desired region.

While, for the most part, the high resolution satellite imagery meets the wide ranging civil and commercial remote sensing exigencies such as environmental monitoring, map making, humanitarian and disaster relief operations, infrastructure planning, traffic control, crops and vegetation monitoring, emergency evacuation planning, it has implications at both national and domestic level. Without any political borders, a broad range of government agencies (including international), the news media, businesses and nongovernmental organizations will gain access to satellite images with the sharpness and quality previously available only from U.S., Russian, and French military satellites. Even individuals and small groups will be able to purchase images off the Internet. As one can imagine, not all the purchasers of this information have benign intentions. Some governments and private groups may exploit this information for aggressive purposes. This may have grave implications on national security and personal privacy. For example, while it is reasonable to expect public entities to observe their citizens or their property, the prospect of an unknown private individual making the same observation from a remote location, perhaps via the web, is a bit disturbing. The threat to privacy is compounded due to the fact that public entities, such as local governments and public utility companies, collect, use and disseminate a large amount of personal information. With the combination of this publicly available personal data pools and the high-resolution image data, it is not impossible to not only identify an individual or an organization by name and address, but can visually “see.” Potential threats include observation of military operations and movements by foreign countries, surveillance of outdoor activities by criminals, tracking of shipping volumes and patterns of a company by its competitors, harmful usage of satellite imagery by terrorists and narcotics criminal groups.

This raises the issue of dealing with both the benign and harmful use of high-resolution imagery. As such, there is a tremendous need to balance public needs and private interests in producing and consuming earth-observation data, as well as reconcile the need for international cooperation with the inevitable pressures arising from commercial competition. As a result, national governments should act as regulators of commercial observation satellites in distributing these imagery products, which calls for controlled dissemination of high-resolution imagery. Policies for prohibiting the release of imagery beyond a certain resolution (such as the guidelines provided by the Department of Commerce), notifying when an image crosses an international boundary, or when such a request is made, etc., are beginning to emerge [Baker01,Wright99]. In fact, commercial organizations such as Space Imaging [space] agree that the image collection or distribution must be constrained where there was an immediate and substantial threat to national security or foreign policy commitment. For example, currently they enforce certain policies to control the dissemination of its imagery. It will not sell imagery to defined terrorist nations and violate UN or bilateral trade restrictions. Moreover, all of its affiliates and re-sellers must comply with applicable restrictions. If directed by the U.S. government the sensor can be ‘turned off,’ coverage over a particular area of the Earth can be limited, or the distribution of imagery itself can be restricted. For example, due to the 1996 Kyl-Bingaman Amendment to the Defense Appropriations Act, an American remote sensing company cannot collect or disseminate imagery of Israel at a better resolution than what is generally available from remote sensing companies in other countries. As a result, images of Israel cannot be distributed at a resolution any higher than two meters.

There is thus a great need for effective and efficient schemes for facilitating such controlled dissemination of satellite imagery and the information products generated from it. Recently, Chun and Atluri [Chun01] have proposed an authorization model to provide access control for geo-spatial images based on their resolution, called GeoSpatial Authorization Model (GSAM). In GSAM, authorizations let one to specify that a subject is allowed to view a specific image or region with a certain resolution, or allowed to overlay a set of images with a specific resolution. To accomplish this, GSAM supports, in addition to *read*, *insert*,

delete and *modify*, other privilege modes such as *view*, *zoom-in*, *overlay* and *identify* that can be defined based on the allowed resolution level for a given subject.

Although implementation of authorizations as access control list, capability list or access matrix is suitable for traditional data, it is not adequate for multimedia data. This is because, as some of the authorizations comprise of non-traditional attributes, such as spatial as in this case. Since authorizations on geospatial imagery involve spatial attributes, it is more efficient to use a spatial indexing structure for managing the authorization base and searching for authorizations based on the spatial extent.

When a user issues a request to access an object(s), specified either as an area or an object identifier, in a specific privilege mode, the system must first verify whether such an authorization exists in the authorization base. Then it must retrieve the relevant objects from the image database. As such, serving an access request requires searching for authorizations from the authorization base, as well as searching for relevant images. That is, once it is verified that a subject can access an object, which is an image, its retrieval requires another index search. We argue in this paper that if we devise an indexing structure for images that is capable of additionally representing authorizations on top of the image index, processing of access requests can be done more efficiently as it is necessary to search only one index structure. In this paper, we propose such a structure, called *RMX-quadtree*, that allows one to implement access control on top of a spatial index. The key idea is to store higher resolution data at lower levels of the tree so that access control can be provided by controlling the traversal of the tree in two directions: (1) By controlling the depth a user can traverse, one can control the resolution of the image (s)he can access. For example, anyone can access a low resolution image such as the New Jersey state map, but access to a 1m x 1m resolution image over a specific sensitive region is restricted. (2) By controlling the breadth a user can traverse, one can control the spatial extent of the image (s)he can view. As an example, a user is given access to the high resolution image (say 1m x 1m), then (s)he is allowed access to only certain regions (probably the property (s)he owns, public parks, etc.), but not to all.

This paper is organized as follows. In section 2, we present the authorization model for geo-spatial imagery and discuss how an authorization index structure can be created. In section 3, we present our proposed indexing structure, the *RMX-quadtree*. In section 4, we demonstrate how RMX-quadtree can index both data and authorizations in a uniform way. In section 5, we present the algorithm to conduct both searching for an image and evaluation of an authorization simultaneously. In section 6, we provide some insight into our current and future research in this area.

2. Geospatial Authorizations and Indexing

2.1 Geospatial Authorization Model

In this section, we review and extend the GeoSpatial Authorization Model (GSAM), proposed earlier by Chun and Atluri [Chun01]. For providing controlled access to geospatial images, GSAM uses publicly available user information, such as property ownership and voter registration records to determine the spatial extent that the user is allowed to access. This in turn is used to determine the appropriate image(s), or a portion of an image covering that region.

Objects in GSAM include geospatial *raster* images with multiple resolutions that represent a geographical region. We assume that each geospatial image object is associated with a unique identifier, *id*, and comprises of spatial features including *latitude* (t), *longitude*(g), *height*(h), *width*(w), *resolution*(r), and a *link*(l), or canonical landmarks like the name of the city or the street. In the latter case, a geographic information system automatically converts this into the corresponding coordinates. In addition to the

raster images that store image as a number of pixels, objects include digital *vector* data that store image as points, lines and polygons, typically, satellite images, digital orthophoto quads and scanned maps are raster images. Every satellite image undergoes a georegistration process that registers it with a known coordinate system. Maps of vector type (e.g. Shape file), digital line graphs, or census TIGER data are vector images. In addition, objects include *tabular* data linked to the images, which includes thematic layer information such as census data, voter registration, land ownership data, and land use data.

As specified in section 1, in addition to the usual *privilege modes* such as insert, delete and update GSAM supports other privileges such *view*, *zoom-in* and *identify* that are required to provide controlled access to multi-resolution imagery. The view privilege allows a user to see an image object covering a certain geographic area, the zoom-in privilege allows a user to view an image covering a certain geographic region at a specific higher resolution, the overlay privilege allows users to generate composite images, where a composite image is constructed by overlaying one image on top of another, and finally, the identify privilege allows the user to view the tabular data linked to an image.

We enhance the subject paradigm by associating them with a set *credentials*, and grouping them into classes (similar to *credential types* [Adam02]). These classes can then be organized into a hierarchical

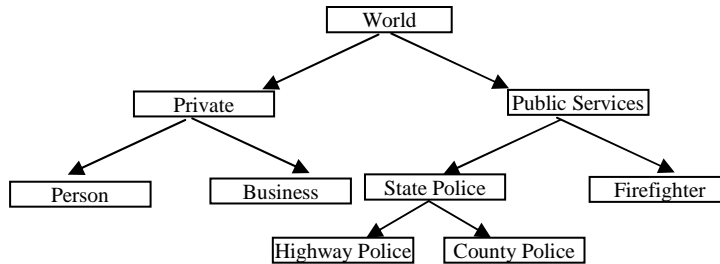


Figure 2: Subject Class hierarchy

structure, as shown in Figure 2. *Subjects* belonging to a class inherit privileges from its super classes. An object in the *authorization* specification can be a single image, a set of images, or a region. Authorizations in GSAM allow one to specify that a subject is allowed to view a specific region with a certain resolution, or allowed to overlay a set of images with a specific resolution. In the following, we formally present authorizations on geo-spatial data. These authorizations are similar to those in GSAM [Chun01], but we extend them with another attribute, called *condition*, which is an expression over subject and object attributes. Due to space limitations, we do not present the formalism adopted in formulating these expressions; we simply explain them with examples in this paper. Furthermore, we allow both negative and positive authorizations. Let $S=\{s_1, s_2, \dots\}$ denote a set of subjects, $C=\{c_1, c_2, \dots\}$, a set of subject classes, $O=\{o_1, o_2, \dots\}$ a set of objects, and $M=\{view, zoom-in, overlay, identify, insert, delete, update\}$ a finite set of privilege modes.

Definition 1: [Authorization]

An authorization a is a tuple $\langle sub, obj, pr, cond, +/- \rangle$ where sub is:

- (i) a subject $s \in S$, or
- (ii) a set of credentials $c \in C$,

obj is (i) an object $o \in O$,
 (ii) a region represented as a rectangle with longitude, latitude, height, width, or
 (iii) a set of object *ids*,

pr is (i) a single privilege mode $m \in M$ or
 (ii) a set of privilege modes $\{m_1, m_2, \dots\} \subseteq M$,

$cond$ is a logical expression, and

+/- indicate it is allow/deny.

The above authorization specification states that a *sub* is allowed (or denied) access to the object(s) *obj*, with privilege *pr* if condition *cond* is satisfied. We use *a.sub* and *a.obj* to denote the specified subject and object in the authorization, respectively. We use *sub.c_i* to denote a credential *c_i* of a subject *sub*. Given an object *o*, we use *region(o)* to represent the region covered by *o*. Valid authorization rules are presented in Example 1.

2.2 Indexing Geospatial Authorizations

Typically authorizations are implemented either as access control lists or capabilities. As can be seen above, authorization specifications on geo-spatial data include spatial attributes (such as objects represented as regions), and privilege modes containing geo-spatial operations (such as zoom-in). Therefore, to efficiently identify the authorizations relevant to an access request, an index structure is necessary. In the following, we present a simple indexing structure for authorizations, and show that such a simple structure is not suitable for a realistic scenario when the number of authorizations is large and distributed within several levels of resolution. Considering the fact that geo-spatial data is organized using spatial index structures, it is prudent to overlay authorizations on top of it. Later in section 4, we present our uniform index structure. Before we construct such a structure for authorizations, the following helpful observations can be made.

(1) Within a single level of resolution, the number of authorizations associated with a region is directly proportional to the spatial extent of the area. In other words, if a subject has the privilege to access to a region, he will also be allowed to access all the regions included in it. If a subject does not have the privilege to an entire region, he may possess an authorization associated with a smaller region. Starting from this assumption, and given the fact that the authorization objects are spatial regions, it is possible to organize the authorizations into a hierarchy based on the area they cover.

Example 1: Consider the following set of authorization rules associated with images within a single level of resolution (1-meter resolution):

$a_1 = \langle \text{Police Officer, NY-City, view:1, none} \rangle$

$a_2 = \langle \text{Private Citizen, Central Park, view:1, none} \rangle$

$a_3 = \langle \text{Private Citizen, Downtown NYC, view:1, } sub.residence = \text{"NY-City"} \rangle$

The above authorizations can be interpreted as follows: a_1 specifies that a Police officer (or subjects belonging to its subclasses) can retrieve all the images within NY-City without any restrictions. a_2 states that any private citizen can access all of the central park region without any restrictions but he/she has to be NY-City resident to retrieve images of downtown (a_3). Based on the spatial extent of the objects in the authorization, these authorizations can be organized into a hierarchical structure, as shown in Figure3.

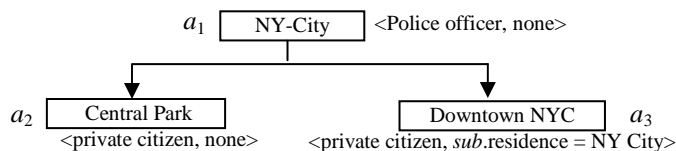


Figure 3 Authorization Hierarchy of Example 1

Consider now that a Newark Police Officer requests access to an image within Central Park; only authorization a_1 , which allows any Police Officer to retrieve an image in NY-City, need to be evaluated. On the other hand, if a private citizen makes the same request, since a_1 is not valid in this case, we need to continue traversing the structure. Since a_2 is true, the user is allowed to retrieve the requested image. However, note that a_3 need not be evaluated in this case. Therefore, organizing the authorizations in a hierarchical structure can considerably reduce the number of authorizations to be evaluated. That is, the

valuation process need to traverse the hierarchical structure only until it finds an authorization that satisfies the access request.

(2) The above concept can be also applied to subject classes as they inherit the privileges of their super-classes.

(3) The last observation applies to the permissions on a region through different levels of resolution. It is reasonable to assume that if a subject is allowed to view a region with a certain level of resolution; he is always allowed to view the same region with a lower resolution. In other words, the zoom-out operation is always allowed.

Based on these assumptions, we can organize the authorizations into a hierarchical structure, where authorizations in the higher levels are associated with lower resolution images covering larger spatial regions, and subjects of more general classes. At lower levels of this structure we will find authorizations associated with higher resolution images covering smaller regions, and subjects belonging to more specific classes.

However, such an index structure may result in inefficient organization (unbalanced trees) when authorizations are not uniformly distributed over the region. For example, consider the image shown in



Figure 4: An Unbalanced distribution of authorizations

Figure 4. The polygon on the left represents a river with a park in the middle, while the region in the dotted polygon on the right includes a residential area with houses, parks, police stations, schools, etc. Considering a single high-resolution level, the left polygon is associated with a single authorization as it is a public place, as everyone is allowed view it. On the other hand, there are different authorizations associated with different parts within the area in the right polygon as more restrictive access constraints apply to this region due to private ownership.

Organizing all these authorizations in a single tree based on the covered region will result in an unbalanced tree (considering also the fact that object regions of authorizations usually overlap). Instead of trying to construct a balanced tree, in this paper, we choose a different strategy, which is to layer authorizations on top of the index tree used for image databases.

3. RMX-Quadtree

In this section, we present our proposed indexing structure called *RMX-quadtree* that not only is capable of storing geo-spatial data organized through different levels of resolution for efficient retrieval, but also allows overlaying of authorizations on top of this index structure. In section 3.1, we review the MX-quadtree, as RMX-quadtree is based on this. In section 3.2, we introduce the structural details of the RMX-quadtree. In section 3.3, we discuss the need for enlarging a region in the RMX-quadtree in order to process insert, delete and search operations that are relevant to process access requests.

3.1 Indexing Structures

In the past three decades, a number of researchers have studied the problem of organizing spatial data, and proposed several multi-dimensional index structures [Gaede97, Ahm97, Subrah98]. Currently, does not exist a spatial indexing structure that is capable of indexing *regions* covered by images based on their *resolution* level. In this paper, we propose a resolution based indexing structure called *RMX-quadtree*, which is an extension of the well-known MX-quadtree. MX-quadtree is a spatial index structure for two-dimensional points. Since our proposed RMX-quadtree is based on the MX-quadtree, we review it briefly below.

MX-quadtree is based on the principle of recursive decomposition; it splits the region into a grid of size $2^k \times 2^k$ where k is the *level of granularity (or level of detail)*. The value of k can be freely chosen before constructing the tree, but has to be kept fixed later. Each node is recursively divided into four equal regions. Given a node N , its four children represent northwest (N.NW), southwest (N.SW), northeast (N.NE) and southeast (N.SE) quadrants of N . All the data objects are stored at the leaf nodes. Figure 5 shows a simple representation of the MX-quadtree where the nodes at first level of the tree represent the four quadrants of the region. Each of these quadrants is further divided into four more quadrants, and so on. Points A, B, C , and D shown in the map on the left are stored based on their position within the region, as shown in the index at right. Insertion and retrieval of an object requires traversal of the tree until the leaf node based on its coordinates.

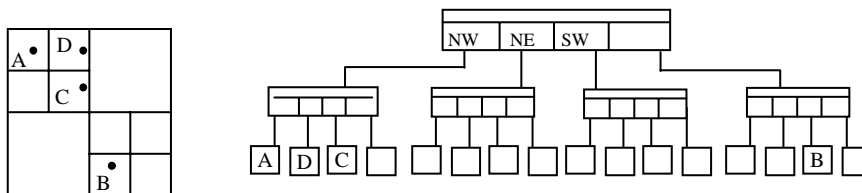


Figure 5: MX-quadtree

The reason why we decided to use the MX-quadtree in building RMX-Quadtree is mainly due to its “deterministic” property, where the “shape” and the height of the tree do not depend on the number of the objects and the order of insertion into the structure. In section 3.2, we will make use of this propriety to extend the MX-quadtree to create RMX-quadtree for indexing multi-resolution images.

3.2 Construction of the RMX-Quadtree

As a first attempt to build a resolution based indexing structure, in this paper, we impose some restrictions on the nature of the images. Later, we intend to relax these assumptions and extend our work to more general cases. We assume that (1) the spatial region represented by each image is a square, (2) images with the same resolution are non-overlapping, (3) images with the same level of resolution have the same dimensions, and (4) higher resolution images imply smaller spatial extent. In addition, we ignore the temporal aspects such as the time at which the image is valid, etc.

As in the MX-quadtrees, the region is split into a grid of size $(2^k \times 2^k)$ where k (the level of detail or granularity) is fixed in advance. The nodes in the RMX-quadtrees represent geospatial regions as opposed to the nodes in MX-quadtrees that represent points. We represent regions by considering an image as a tuple $\langle center, side \rangle$, where $center$ and $side$ represent the center and the edge of the image, respectively. By constructing it this way, images can now be represented as points, and therefore can be easily indexed, just as in the MX-insert process. Clearly, when processing the query requests one must consider the fact that we are representing areas, but not points, and return the appropriate region that satisfies the query (explained in section 5).

Given an image i , we use l_i to represent its resolution level and p_i to represent its side. Based on our third assumption stated above, p_i is a function of l_i . The intuitive idea behind the RMX-Quadtree is as follows. Assume we group the images with the same resolution, and conceptually construct a MX-quadtrees for each group of images separately. The RMX-quadtrees is then constructed by superimposing these independent MX-quadtrees into one by joining them at the appropriate levels. We elaborate this process below.

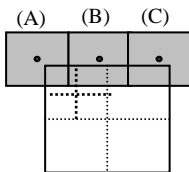
The height (k) of each of these trees can be computed as follows. Given a single level of resolution l_i representing non-overlapping images of side p_i , it is easy to demonstrate that the maximum number of images representing the region is $num_i = (\frac{n}{p_i} + 1)^2$ where n is the side of the square representing the entire spatial region of interest. Since the lower levels of the tree are generated by splitting the node into 4 equal-size regions, we need to have the number of partitions (k_i) at least equal to the 4th root of the total number of images. Formally:

Definition 2: [Level of granularity k_i]

Given the level of resolution l_i , k_i represents the minimum level of granularity to guarantee that the MX-quadtrees contains at most one image per leaf.

$$k_i = \lceil \sqrt[4]{num_i} \rceil = \lceil \sqrt{\left(\frac{n}{p_i} + 1\right)} \rceil$$

The assumption that k_i generates an MX-quadtrees, where each leaf is associated with at most one image of size p_i , is feasible because images are represented using their center (but not the region covered by the image). The same assumption could not be met if upper-right or lower-left corner were used instead of the center. For example, from the figure on the left, we can see that by using the upper-right corner of the images maps two objects (A and B) into the same node (the most north-west leaf). Instead, if we use the center of the image, A, B and C will fall into different leaves as desired. One can formally prove that the same is true if one uses other corners, by comparing the minimum distance between two images and the side of each leaf. Using k , one can construct a tree for each set of images with the same level of resolution.



The node structure of RMX-quadtrees is as follows:

```

mrqtnode = Record
    Info: imagetype;
    Xval,Yval: real;
    Resolution: real;
    Metadata: metadatatype;
    NE, NW, SE, SW: ↑mrqtnode;
    XLB, XUB, YLB, YUB: integer + {-∞,∞}

```

In the above, Info represents a link to an image, Xval and Yval the geospatial coordinates of its center, resolution the side of the image, the tabular information associated with the image, NE, NW, SE, SW the 4 children as in the original MX-quadtrees, and XLB, XUB, YLB, YUB the spatial extent (or bound) represented by the node.

Given that an MX-quadtrees sequentially splits the region into four equal-size areas independent of the inserted data, it is possible to combine all the MX-quadtrees for each level of resolution into a single RMX-quadtrees. The resulting tree will have a height equal to the k_i of tree for the set of images with the highest level of resolution, i.e., the maximum of all k_i . At each level, the images are associated with the appropriate level of the RMX-quadtrees. We show how this can be done with the following example.

Example 2: Considering a simple scenario where the region is represented by images downloaded from two different satellites, Sat1 and Sat2, with different resolution levels. Assume Sat1 will have $k_1=2$ and Sat2 will have $k_2=4$ (each k_i is calculated according to the formula above). Figures 6 (A) and (B) show the MX-quadtrees for the image sets of SatA and SatB, respectively. The gray squares represent leaves containing images.

Now, these two MX-quadtrees representing images from two different satellites can be merged into a single RMX-quadtrees, as shown in Figure7(C). The resulting RMX-quadtrees will have height equal to $k_{\text{rmx}}=\max(k_1,k_2)=4$ and images stored in the intermediary nodes other than the leaves, i.e. in nodes at level $k=2$.

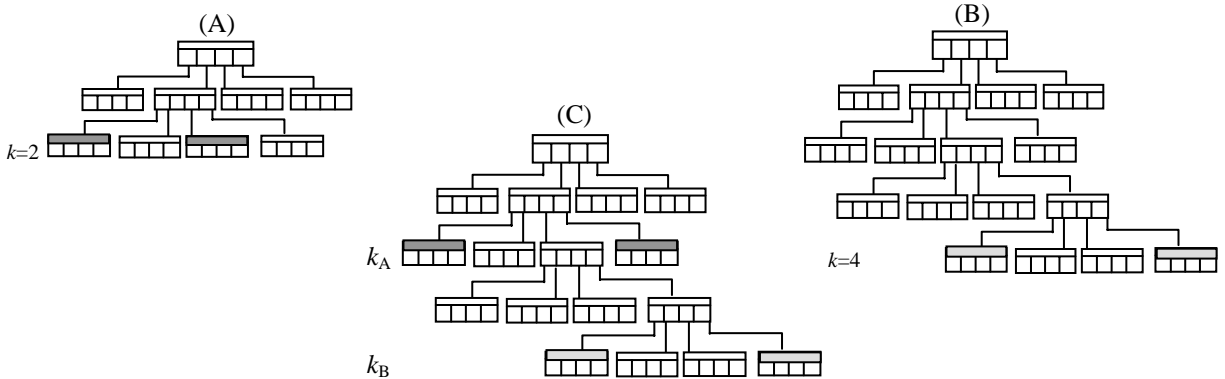


Figure 6: Merging different MX-quadtrees into a single RMX-quadtrees

3.3 Region Enlargement

Representing regions as points in RMX-quadtrees does not change the procedures for *inserting* and *deleting* images used for MX-quadtrees. The only difference is that in RMX-quadtrees the process has to traverse the tree only until the level corresponding to the k_i associated with the image, and then insert or delete it from the database. However, transformation of regions to points makes *searching* for an image, and therefore *evaluating an access request*, more complex. The image requested by a user's query covering a specific region with a certain level of resolution may (i) *exactly* match an image, (ii) *contained*

in an image, or (iii) *span* several images stored in the database. We refer the first two cases as an *exact query* as a specific image needs to be retrieved. Since the third case requires a number of images, we consider this as a special case of a *range-query*. In other words, a query may require merging of different images, or retrieving only a portion of an image or an entire image. An additional complication due to the transformation of region to points is that the region covered by an image stored in a node might not correspond to the area represented by the node (unlike in MX-quadtrees where the data stored in the leaves are points in the region represented by the node). We explain this problem with the following example.

Example 3: Assume the black square (in Figure 7) represents the region the user wants to retrieve. It is possible that this black square is either part of image A (gray region) located in the NW subtree of the root, or part of image B (gray region) stored in the NE subtree of the root. Since we do not know in advance where the images are stored, at a certain point in the search process, it is necessary to traverse different subtrees.

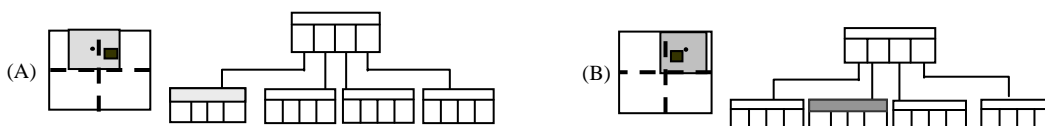


Figure 7: The region of interest may belong to different subtrees

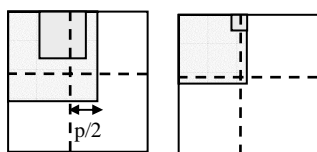


Figure 8: Enlarged region

As a result, the cost for searching increases. The reason for this ambiguity is because we construct the index based on points rather than regions. However, if we can compute in advance the maximum possible region covered by the node in a subtree, we can restrict the search between multiple subtrees. Essentially, we adopt the following strategy to minimize the cost of search.

We enlarge the bounds of the area represented by a node, and choose a subtree path only if the enlarged area overlaps the region we want to retrieve. This operation depends upon the side of the images we are searching; the bigger the side, the wider the “possible area” covered by images in the nodes. We formally define the enlarged area as follows.

Definition 3: [p_i -enlarged region of N]

Given a node N and image of side p_i associated with a level or resolution l_i , the p_i -enlarged region of N, denoted by $Enl_{p_i}(N)$, is the spatial region obtained by extending N’s bounds by $p_i/2$.

Figure 8 shows two different sizes of images (the gray squares) and how the area related to a node (NW child of the root) has to be enlarged to prevent the problem presented above.

4. A Uniform Index Structure for Geo-spatial Data and Authorizations

In this section, we show how authorizations can be layered on top of the RMX-quadtrees. Before we proceed with the details, it is important to remark the relation between images and the authorizations. A single authorization specification may apply for several images, as well as one image may have several authorizations applicable to it. As in Example 1, authorization $a_1 = \langle \text{police officer, NY City, view:1, none} \rangle$ is applicable to every image covering the NY-City area. On the other hand, if we consider a very high-resolution image showing an area with a park and many private residences, that image is probably associated with a number of different authorizations based on rules such as public availability of the park and allowing only the owners (or tenants) to view the area of residence.

To directly map the authorizations onto the RMX quadtree, we layer all the authorizations related to each level of resolution l_i within the nodes at the corresponding level of detail k_i . This way, while searching for images, one has to first reach the level containing images, then evaluate the authorizations associated with the area represented by that node, and finally retrieve the image if the user is authorized. If an authorization refers to an area represented by several images, it needs to be placed in each node overlapping that area. Although this solution is better (in terms of number of authorizations to evaluate) than simply using an access control list, it can be further improved. Instead of layering the authorizations applicable only to a node at a certain level k , we also store them at all its ancestors' nodes. When computing the area represented by the node, we must also take into account the enlarged-region. The following definition formalizes the condition that must be satisfied to store an authorization at a node.

Definition 4: [Spreading rule]

Given the level of resolution l_i , and side p_i of the image(s) i stored at node N of the RMX-Quadtree, an authorization $a = \langle sub, obj, pr, cond, +/- \rangle$ is stored in N , if the enlarged-region $Enl_{p_i}^l(N)$ is totally included in $region(a.obj)$ and $l_{a.obj} = l_i$.

In other words, an authorization is stored into a node N when it applies to all the possible images in N 's subtrees. Spreading rule is based on the concept introduced in section 2 that if a subject is allowed to access an area, he is also allowed to access all the regions at the same level of resolution within it. In the RMX-Quadtree, the area covered by a node N with a given level of resolution represents the enlarged area generated according to Definition 3.

Example 4: Figure 9 shows a simple example where the object-area (gray rectangle) covers the entire enlarged-area (dotted square) of the NW child of the root. In this case, according to our spreading rule, the authorizations applied to all the images stored in its subtrees can be directly stored in the NW node (gray colored).

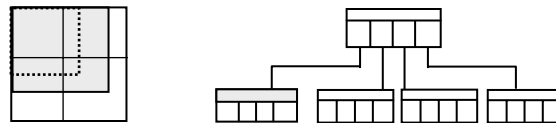


Figure 9: Spreading Rule in the RMX-Quadtree

The spreading rule process has to be blocked when it reaches the level of detail the authorization refers to. Going deeper in the tree will not do any good as the process of retrieving images stops when it reaches the level containing the images of that resolution. The solution above is suitable for cases where one authorization is applicable to one or more images. When an authorization is applicable to only part of an image, there are no nodes in the tree that satisfy the spreading rule. However, we still do not want these authorizations to be stored at levels deeper than the one containing images, as lower levels refer to images of higher resolution. To address this issue, we store all such authorizations in a separate spatial structure, called AUTH-quadtree, which can be linked to this node (similar to the MX-CIF quadtree [Samet90A, Samet90B]). The following spread-rule algorithm details these steps.

```

Algorithm 1: {Spread_rule}
/* Spread an authorization on the RMX-Quadtree
Input: RMX-quadtree, N = Root of the tree
      a = < sub,obj,pr,cond,+/- >
      p = side of the image (level of resolution) of obj */
//var x: number of levels traversed (initial value is 0)
//var T: enlarged area covered by Node
compute k, level of granularity for the given p;

```

```

Procedure: Spread_rule(N, a, p, x)
{
T.XLB = N.XLB - p/2; T.XUB = N.XUB + p/2;
T.YLB = N.YLB - p/2; T.YUB = N.XLB + p/2;
if (T) ∩ region(a.obj) = ∅ then Halt; //(Case 1)
if (T) ⊆ region(a.obj) then //Store the authorization in the node (not leaf)
    {
    Store a at N of RMX-quadtrees; //(Case2)
    Halt;
    }
if (k=x) then //search has reached the leaves of the tree (Case 3)
    {
    if N.image ∩ region(a.obj) then store a in AUTH-quadtrees;
    Halt;
    }
x = x+1 //Further Traversal is required (Case 4)
Spread_rule (N.NE, a, p, x);
Spread_rule (N.NW, a, p, x);
Spread_rule (N.SE, a, p, x);
Spread_rule (N.SW, a, p, x);
}

```

Algorithm *Spread_rule* accepts as input the authorization (expressed as tuple $\langle sub, obj, privilege, cond, Rule, +/- \rangle$), a node of the tree N and the side of the object in the authorization. x is a variable that counts the number of levels traversed. The algorithm traverses the tree and for each node compares the authorization object with the node's enlarged-area. One of the following 4 cases is possible:

Case1: The node's enlarged-area does not overlap the authorization object's region. Then the process stops, and the authorization is not saved at that node.

Case2: The node's enlarged-area is included in that of the object. If the spreading rule is verified and the rule is stored in the node, then the process ends because the authorization already applies to all the subtrees of the node.

Case3: The process has reached the level of resolution of the authorization object, and if the area covered by the rule overlaps that of the image stored in the node, the authorizations will be stored in the AUTH-quadtrees linked to the node.

Case 4: All the previous conditions are not satisfied meaning that the region of the authorization object overlaps the area covered by the node, but the spreading rule is not verified. Then the process will be repeated with all the four subtrees of the node until one of the above three cases is reached.

In summary, if the region covered by the object includes the region covered by the node, then we store the authorization in the RMX-quadtrees itself. On the other hand, if the region covered by the authorization overlaps with the region of the image stored in N , and we are at the leaf node, then the rule is stored in the AUTH-quadtrees linked to the node.

To accommodate such layering of authorizations and AUTH-quadtrees, we need to enhance the node structure of the RMX-quadtrees. We extend the node structure with the vector $\langle level, rule \rangle$ where *level* represents the level of resolution and *rule* the authorization associated with the node or the link to the root of the AUTH-quadtrees in case nodes containing multiple authorization objects. We do not provide further details of this in this paper.

5. Combined Image Search and Authorization Evaluation

In the previous section, we have presented *spread_rule*, the process for mapping the authorizations at appropriate nodes in the RMX-quadtrees. As a result, a uniform indexing structure now stores both geospatial images and authorizations. In this section, we present the algorithm for processing an access request, which simultaneously searches for the image(s) that satisfy the request, identifies the applicable authorizations, and evaluates them. Graphical interfaces facilitate specification of access requests in many ways, ranging from specifying an image Id or a region of interest, clicking on an already viewing image to view at a higher resolution by further zooming-in, specifying the region with its name, etc. We assume the access request $ar = \langle sub, obj, l_{obj}, pr \rangle$. We deal with the case of zoom-in request in the next subsection.

From this access request, the search_evaluate process computes the level of granularity (k) based on the resolution, the side of the image (p) for being able to enlarge the node-area, the geospatial region the user wishes to retrieve ($region(obj)$). Then the process traverses the RMX-quadtrees starting from its root (N). In addition to searching for the images, at each step the process identifies and evaluates the authorizations applicable to the access request.

The search_evaluate process takes one of the following three cases: (i) Stops evaluating the authorizations. This occurs when N contains an authorization that allows the user to retrieve a wider area. That is, the user, in fact, is allowed access to all the images in N 's subtree, and therefore, no more authorizations need to be evaluated. But, the search process continues by traversing the tree for reaching the level containing the relevant image(s). (ii) Keep traversing the tree and check for more authorizations until reaching the node on level k , and (iii) Halt the process completely. This occurs when a negative authorization is applicable at a non-leaf node N or when successful with the traversal in case (ii).

Algorithm 2: {Search_Evaluate}

/* Processing an access request

Input: Access request $ar = \langle sub, obj, l_{obj}, pr \rangle$

RMX-quadtrees, $N = \text{Root of the tree}$ */

//var x : the number of levels traversed (initial value is 0)

//var T : enlarged area covered by the node

//var Access: access status (initial value is undetermined)

compute k , the level of granularity of the tree;

compute p , side of the image from l_{obj} ;

compute $region(obj)$;

Procedure: Search_Evaluate(N, ar, p, x)

{

$T.XLB = N.XLB - p/2$; $T.XUB = N.XUB + p/2$;

$T.YLB = N.YLB - p/2$; $T.YUB = N.YUB + p/2$;

if $T \cap region(N) = \emptyset$ then Halt;

if ($k = x$) then //search reached the leaves of the tree

{

if $region(N) \cap region(obj)$ then

{

if (Access == undetermined) Access = evaluate_auth_leaf(N, A); //search in the AUTH-
//quadtree

if (Access == allow) return image

Halt;

}

}

```

if (Access ==undetermined) Access = evaluate_auth(N,T); //evaluate the authorizations in the node
//of the tree
if (Access==deny) Halt;
x = x+1 //The process will continue evaluating the next level of the tree
Search_Evaluate (N.NE, ar, p, x);
Search_Evaluate (N.NW, ar, p, x);
Search_Evaluate (N.SE, ar, p, x);
Search_Evaluate (N.SW, ar, p, x);
}

```

While *Evaluate_auth* is a procedure for checking the authorization stored in a non-leaf node itself, *Evaluate_auth_leaf* is a procedure to traverse the AUTH-Quadtree attached to the node at level k . Both these procedures return ‘deny’ if the user is not allowed to access the image (negative authorization case only), ‘allow’ if the access is allowed, and ‘undetermined’ otherwise (meaning that the process need to evaluate more authorizations to allow or deny access to the image). If the process *Evaluate_auth_leaf* returns ‘undetermined’ means that there are no authorizations for the subject associated with the requested region. Since this search process checks for the authorizations early on in the tree, the number of authorizations to be verified is significantly low. In this paper, we do not deal with the issue of conflict resolution [Bert93,Adam02].

The output of the search_evaluate procedure will be all the images overlapping the area requested by the user. However, we realize that after retrieving these images, they must be either merged or cropped before displaying to the user the region he is authorized to view and requested by the user as a single picture. We do not discuss this post-processing of images in this paper.

5.1 Zoom-in Access Request

In addition to serving access requests to view images, a multi-resolution image database is required to serve zoom-in requests. In other words, a user may click further on an image to retrieve more detailed information about the image. Although it is possible to make use of specific algorithms for zooming-in an area, after a certain level of zoom-in, the quality of the images decreases and the information becomes useless. Therefore, to provide the zoom-in functionality without reducing the quality, we need to provide images with higher resolution levels. Such zoom-in access requests require furthering the search within the tree until reaching the level containing the images with higher-resolution. One way to accomplish this is, however, to start a completely new search with a different value for k and p . However, by servicing a zoom-in as a new access request seems to be not a good idea because the search process has already traversed part of the RMX-Quadtree. One might think that by simply traversing the tree further down would suffice to serve such zoom-in requests. However, in the following, we explain why it is not as simple as that.

To better explain the problems encountered with the zoom-in operation, we first consider the case where the region retrieved on lower-resolution is contained in one single image. In other words, we do not consider (at this point) regions that result in merging several images. We assume N is the node containing the image when the user requests for a zoom-in. Based on the results obtained from the search_evaluate image, N represents area the enlarged with $p/2$ side wider than the one represented by its bounds, shown in the dotted circle in Figure 10(a). In Figure 10(b), we present in detail the enlarged area in the circle and show the bounds of the region represented by N and the extended region. In case of zoom-in to a different level of resolution, the side of the image would be different, say q , and is always less than p . The extended area for N is shown in Figure 10(c).

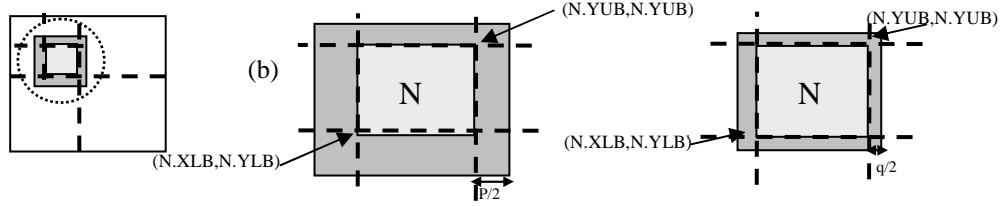


Figure10: Enlarged area for two levels of resolution

Figure 11 represents the overlap of the two extended regions of Figures 10(b) and (c), from which we can extract the information regarding where the higher resolution image can be found in the tree. Let w be the side of the node N and x,y its center. Based on the location of the zoom-in region, one of the following three cases is possible.

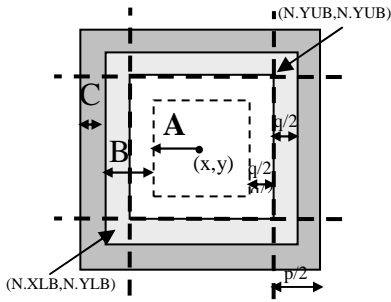


Figure 11: Zoom-in

(i) *Totally included:* If the zoom-in region is included in the area with side $(w-q/2)$ and center (x,y) (A), then it means that the higher resolution image is located in a subtree of N . In other words, a zoom-in request for a region in this area can be served simply by the search_evaluate process with N as root and q as image-side.

(ii) *Partially included:* If the zoom-in is within the region with center (x,y) and side $(w+q/2)$ (excluding the area represented by A), retrieval of higher resolution images requires searching not only N 's subtrees, but also traversing some of the subtrees of N 's father. This is because this area represents the overlapping area between two different nodes for images with side q .

(iii) *Not included:* If the zoom-in region is in C, then the higher resolution images cannot be found traversing N 's subtree, as they are somewhere else in the tree. They can be found only by recursively executing the search_evaluate procedure with N 's father as the root, and its father, and so on.

While in the first two cases it is efficient to continue the traversal from N to serve zoom-in requests, it is prudent to process as a new access request if it happens to fall in the third case. The problem is more complex when the region is already the composition of different images. In this case, the procedure must be repeated for each of those images and also must take into account avoiding searching the same branch of the tree more than once. We do not discuss further these details in this paper.

6. Conclusions and Future Work

In this paper, we argue that commercially available high-resolution satellite imagery has both benign as well as malicious implications. The exploitation of such information by some governments and private groups will pose clear threats to national security and privacy. In this paper, we present an authorization model for geo-spatial image databases for facilitating controlled dissemination of satellite imagery and the information products generated from it. Since authorizations on geospatial imagery involve spatial attributes, managing the authorization base and searching for authorizations based on the spatial extent require a spatial indexing structure. For providing effective and efficient means to serve access requests, we propose a uniform indexing structure, called *RMX-quadtree*, which is capable of indexing multi-resolution satellite imagery and additionally allow representing authorizations on top of it. We demonstrate how authorizations with privilege modes such as *view* and *zoom-in* can be implemented. We are currently extending our work to include *overlay* and *identify* privilege modes.

Although implementation of authorizations as access control list, capability list or access matrix is suitable for traditional data, it is not adequate for multimedia data, as some of the authorizations comprise

of non-traditional attributes, such as spatial as in this case. Similar is the case with temporal databases, moving object databases, etc. In future, we intend to explore developing uniform indexing schemes for such data. In addition, we will enhance the RMX-quadtrees to capture the temporal attributes of the geospatial data and authorizations. Furthermore, we intend to conduct a performance evaluation to demonstrate that our uniform indexing scheme indeed has significant impact on the response time. Furthermore, we intend to relax the several assumptions made in section 3.2 in building the uniform index structure.

7. References

- [Adam02]
N.R. Adam, V. Atluri, E. Bertino and E. Ferrari, "A Content-based Authorization Model for Digital Libraries," IEEE Transactions Knowledge and Data Engineering, Volume 13, Number 4, 2002, pages 705-716.
- [Ahn97]
H. Ahn, N. Mamoulis and H. Wong. "A Survey on Multidimensional Access Methods, Technical report", Utrecht University (Netherlands), 1997
- [Baker01]
J. Baker, K. O'Connell and R. Williamson "Commercial Observation Satellites" Rand Edition, 2001
- [Barc00]
T. Barclay, J. Gray and D. Slutz, "Microsoft TerraServer: a spatial data warehouse," Proceedings of the 2000 ACM SIGMOD on Management of data, pages 307-318.
- [Bert93]
E. Bertino, P. Samarati, and S. Jajodia. "Authorizations in relational database management systems". In Proc. First ACM Conference on Computer and Communications Security, Fairfax, VA, November 1993.
- [CBERS]
<http://www.inpe.br/programas/cbers/english/satelite.html>
- [Chun01]
S. Chun and V. Atluri, "Protecting Privacy from Continuous High-resolution Satellite Surveillance" Data and Application Security: Developments and Directions, (eds.) Bhavani Thuraisingham, R. Van de Riet, K. R. Dittrich and Z. Tari, Kluwer Academic Publishers, 2001
- [EROS] <http://www.satellus.se>
- [Gaede98]
V. Gaede and O. Gunther, "Survey on Multidimensional Access methods" ACM computing surveys, 30 (2): 170-230, 1998.
- [OrbView3]
http://www.orbimage.com/corp/orbimage_system/ov3/
- [QuickBird]
<http://www.digitalglobe.com>
- [Samet90A]
H. Samet. "The design and Analysis of Spatial Data Structures" Addison-Wesley, Reading, Mass 1990
- [Samet90B]
H. Samet. "Application of Spatial Data Structures" Addison-Wesley, Reading, Mass 1990
- [space]
<http://www.spaceimaging.com>
- [Subrah98]
V.S. Subrahmanian. "Principles of Multimedia Database Systems", Morgan Kaufmann, January 1998.
- [Wright99]
R. Wright "Private Eyes" The New York Times Magazine, September 1999.