Digital Image Processing

Ming Jiang

School of Mathematical Sciences
Peking University

ming-jiang@pku.edu.cn

May 18, 2010

Outline I

Levels of Representation

Traditional Image Data Structures
- Matrices
- Chains
- Topological Data Structures
- Relational Structures

Hierarchical Data Structures
- Pyramids
- Quadtrees

Data and Algorithm

- Data and an algorithm are the two basic parts of any program.
- Computer program $=$ data $+$ algorithm.
- Data organization can considerably affect the simplicity of the selection and the implementation of an algorithm.
- The choice of data structures is fundamental when writing a program.

Data and Algorithm

- Data and an algorithm are the two basic parts of any program.
- Computer program $=$ data $+$ algorithm.
- Data organization can considerably affect the simplicity of the selection and the implementation of an algorithm.
- The choice of data structures is fundamental when writing a program.

Data and Algorithm

- Data and an algorithm are the two basic parts of any program.
- Computer program $=$ data $+$ algorithm.
- Data organization can considerably affect the simplicity of the selection and the implementation of an algorithm.
- The choice of data structures is fundamental when writing a program.

Data and Algorithm

- Data and an algorithm are the two basic parts of any program.
- Computer program $=$ data $+$ algorithm.
- Data organization can considerably affect the simplicity of the selection and the implementation of an algorithm.
- The choice of data structures is fundamental when writing a program.

Inappropriate Representation for Images

- The difficulties in image processing, image analysis and computer vision come from the bad representation or organization of the data involved.
- In fact, the visual information representation and organization inside human brain is not well understood at present.
- Although we are to discuss some representations used so far, none of them are appropriate for a general purpose processing target.

Inappropriate Representation for Images

- The difficulties in image processing, image analysis and computer vision come from the bad representation or organization of the data involved.
- In fact, the visual information representation and organization inside human brain is not well understood at present.
- Although we are to discuss some representations used so far, none of them are appropriate for a general purpose processing target.

Inappropriate Representation for Images

- The difficulties in image processing, image analysis and computer vision come from the bad representation or organization of the data involved.
- In fact, the visual information representation and organization inside human brain is not well understood at present.
- Although we are to discuss some representations used so far, none of them are appropriate for a general purpose processing target.

Levels of Representation

- The aim of computer visual perception is to find a relation between an input image and the models of the real world.
- During the transition from the raw image to the model, `semantic knowledge` about the interpretation of image data is used more and more.
- Several levels of visual information representation are defined on the way between the input image the model.

Levels of Representation

- The aim of computer visual perception is to find a relation between an input image and the models of the real world.
- During the transition from the raw image to the model, `semantic knowledge` about the interpretation of image data is used more and more.
- Several levels of visual information representation are defined on the way between the input image the model.

Levels of Representation

- The aim of computer visual perception is to find a relation between an input image and the models of the real world.
- During the transition from the raw image to the model, `semantic knowledge` about the interpretation of image data is used more and more.
- Several levels of visual information representation are defined on the way between the input image the model.

Levels of Representation and Algorithms

- Computer vision then comprises a design of the
  - Intermediate representations (data structures).
  - Algorithms used for the creation of representation and introduction of relations between them.

Levels of Representation and Algorithms

- Computer vision then comprises a design of the
  - Intermediate representations (data structures).
  - Algorithms used for the creation of representation and introduction of relations between them.

Levels of Representation and Algorithms

- Computer vision then comprises a design of the
  - Intermediate representations (data structures).
  - Algorithms used for the creation of representation and introduction of relations between them.

Four Levels of Representations

- The representation can be stratified in four levels.
  - Iconic images
  - Segmented images
  - Geometric representations
  - Relational models

Four Levels of Representations

- The representation can be stratified in four levels.
  - Iconic images
  - Segmented images
  - Geometric representations
  - Relational models

Four Levels of Representations

- The representation can be stratified in four levels.
  - Iconic images
  - Segmented images
  - Geometric representations
  - Relational models

Four Levels of Representations

- The representation can be stratified in four levels.
  - Iconic images
  - Segmented images
  - Geometric representations
  - Relational models

Four Levels of Representations

- The representation can be stratified in four levels.
  - Iconic images
  - Segmented images
  - Geometric representations
  - Relational models

Discussions on Four Levels of Representations

- However, there are no strict borders between them and a more detailed classification of the representational levels may be used in some applications.
- For some specific uses, some representations can be omitted.
- These four representational levels are ordered from signals at low level of abstraction to the description that a human can understand.
- The information flow between the levels may be bi-directional.

Discussions on Four Levels of Representations

- However, there are no strict borders between them and a more detailed classification of the representational levels may be used in some applications.
- For some specific uses, some representations can be omitted.
- These four representational levels are ordered from signals at low level of abstraction to the description that a human can understand.
- The information flow between the levels may be bi-directional.

Discussions on Four Levels of Representations

- However, there are no strict borders between them and a more detailed classification of the representational levels may be used in some applications.
- For some specific uses, some representations can be omitted.
- These four representational levels are ordered from signals at low level of abstraction to the description that a human can understand.
- The information flow between the levels may be bi-directional.

Discussions on Four Levels of Representations

- However, there are no strict borders between them and a more detailed classification of the representational levels may be used in some applications.
- For some specific uses, some representations can be omitted.
- These four representational levels are ordered from signals at low level of abstraction to the description that a human can understand.
- The information flow between the levels may be bi-directional.

Iconic Images

- Iconic images — consists of images containing original data; integer matrices with data about pixel brightness.
- E.g., outputs of pre-processing operations (e.g., filtration or edge sharpening) used for highlighting some aspects of the image important for further treatment.

Iconic Images

- `Iconic images` — consists of images containing original data; integer matrices with data about pixel brightness.
- E.g., outputs of pre-processing operations (e.g., filtration or edge sharpening) used for highlighting some aspects of the image important for further treatment.

Segmented Images

- `Segmented images` — parts of the image are joined into groups that probably belong to the same objects.
- E.g., the output of the segmentation of a scene with polyhedrons is either line segments coinciding with borders or two-dimensional regions corresponding with faces of bodies.
- It is useful to know something about the application domain while doing image segmentation; it is then easier to deal with noise and other problems associated with erroneous image data.

Segmented Images

- `Segmented images` — parts of the image are joined into groups that probably belong to the same objects.
- E.g., the output of the segmentation of a scene with polyhedrons is either line segments coinciding with borders or two-dimensional regions corresponding with faces of bodies.
- It is useful to know something about the application domain while doing image segmentation; it is then easier to deal with noise and other problems associated with erroneous image data.

Segmented Images

- `Segmented images` — parts of the image are joined into groups that probably belong to the same objects.
- E.g., the output of the segmentation of a scene with polyhedrons is either line segments coinciding with borders or two-dimensional regions corresponding with faces of bodies.
- It is useful to know something about the application domain while doing image segmentation; it is then easier to deal with noise and other problems associated with erroneous image data.

Geometric Representations

- `Geometric representations` — hold knowledge about 2D and 3D shapes.
- The quantification of a shape is very difficult but very important.
- It is the inverse problem of computer graphics.

Geometric Representations

- `Geometric representations` — hold knowledge about 2D and 3D shapes.
- The quantification of a shape is very difficult but very important.
- It is the inverse problem of computer graphics.

Geometric Representations

- `Geometric representations` — hold knowledge about 2D and 3D shapes.
- The quantification of a shape is very difficult but very important.
- It is the inverse problem of computer graphics.

Relational Models

- Relational models - give the ability to treat data more efficiently and at a higher level of abstraction.
- A priori knowledge about the case being solved is usually used in processing of this kind.
- Example - counting planes standing at an airport using satellite images
  - position of the airport (e.g., from a map).
  - relations to other objects in the image (e.g., to roads, lakes, urban areas).
  - geometric models of planes for which we are searching.
  - etc.
- AI techniques are often explored.
- Information gained from the image may be represented by semantic nets or frames.

Relational Models

- Relational models - give the ability to treat data more efficiently and at a higher level of abstraction.
- A priori knowledge about the case being solved is usually used in processing of this kind.
- Example - counting planes standing at an airport using satellite images
  - position of the airport (e.g., from a map).
  - relations to other objects in the image (e.g., to roads, lakes, urban areas).
  - geometric models of planes for which we are searching.
  - etc.
- AI techniques are often explored.
- Information gained from the image may be represented by semantic nets or frames.

Relational Models

- `Relational models` - give the ability to treat data more efficiently and at a higher level of abstraction.
- A priori knowledge about the case being solved is usually used in processing of this kind.
- Example - counting planes standing at an airport using satellite images
  - position of the airport (e.g., from a map).
  - relations to other objects in the image (e.g., to roads, lakes, urban areas).
  - geometric models of planes for which we are searching.
  - etc.
- AI techniques are often explored.
- Information gained from the image may be represented by semantic nets or frames.

Relational Models

- `Relational models` - give the ability to treat data more efficiently and at a higher level of abstraction.
- A priori knowledge about the case being solved is usually used in processing of this kind.
- Example - counting planes standing at an airport using satellite images
  - position of the airport (e.g., from a map).
  - relations to other objects in the image (e.g., to roads, lakes, urban areas).
  - geometric models of planes for which we are searching.
  - etc.
- AI techniques are often explored.
- Information gained from the image may be represented by semantic nets or frames.

Relational Models

- `Relational models` - give the ability to treat data more efficiently and at a higher level of abstraction.
- A priori knowledge about the case being solved is usually used in processing of this kind.
- Example - counting planes standing at an airport using satellite images
  - position of the airport (e.g., from a map).
  - relations to other objects in the image (e.g., to roads, lakes, urban areas).
  - geometric models of planes for which we are searching.
  - etc.
- AI techniques are often explored.
- Information gained from the image may be represented by semantic nets or frames.

Relational Models

- `Relational models` - give the ability to treat data more efficiently and at a higher level of abstraction.
- A priori knowledge about the case being solved is usually used in processing of this kind.
- Example - counting planes standing at an airport using satellite images
  - position of the airport (e.g., from a map).
  - relations to other objects in the image (e.g., to roads, lakes, urban areas).
  - geometric models of planes for which we are searching.
  - etc.
- AI techniques are often explored.
- Information gained from the image may be represented by semantic nets or frames.

Relational Models

- `Relational models` - give the ability to treat data more efficiently and at a higher level of abstraction.
- A priori knowledge about the case being solved is usually used in processing of this kind.
- Example - counting planes standing at an airport using satellite images
  - position of the airport (e.g., from a map).
  - relations to other objects in the image (e.g., to roads, lakes, urban areas).
  - geometric models of planes for which we are searching.
  - etc.
- AI techniques are often explored.
- Information gained from the image may be represented by semantic nets or frames.

Relational Models

- `Relational models` - give the ability to treat data more efficiently and at a higher level of abstraction.
- A priori knowledge about the case being solved is usually used in processing of this kind.
- Example - counting planes standing at an airport using satellite images
  - position of the airport (e.g., from a map).
  - relations to other objects in the image (e.g., to roads, lakes, urban areas).
  - geometric models of planes for which we are searching.
  - etc.
- AI techniques are often explored.
- Information gained from the image may be represented by semantic nets or frames.

Relational Models

- `Relational models` - give the ability to treat data more efficiently and at a higher level of abstraction.
- A priori knowledge about the case being solved is usually used in processing of this kind.
- Example - counting planes standing at an airport using satellite images
  - position of the airport (e.g., from a map).
  - relations to other objects in the image (e.g., to roads, lakes, urban areas).
  - geometric models of planes for which we are searching.
  - etc.
- AI techniques are often explored.
- Information gained from the image may be represented by semantic nets or frames.

Traditional Image Data Structures

- Traditional image data structures, such as
  - matrices
  - chains
  - graphs
  - lists of object properties
  - relational databases
  - etc.

  are important not only for the direct representation of image information, but also a basis of more complex hierarchical methods of image representation.

Traditional Image Data Structures

- Traditional image data structures, such as
  - matrices
  - chains
  - graphs
  - lists of object properties
  - relational databases
  - etc.

  are important not only for the direct representation of image information, but also a basis of more complex hierarchical methods of image representation.

Traditional Image Data Structures

- Traditional image data structures, such as
  - matrices
  - chains
  - graphs
  - lists of object properties
  - relational databases
  - etc.

  are important not only for the direct representation of image information, but also a basis of more complex hierarchical methods of image representation.

Traditional Image Data Structures

- Traditional image data structures, such as
  - matrices
  - chains
  - graphs
  - lists of object properties
  - relational databases
  - etc.

  are important not only for the direct representation of image information, but also a basis of more complex hierarchical methods of image representation.

Traditional Image Data Structures

- Traditional image data structures, such as
  - matrices
  - chains
  - graphs
  - lists of object properties
  - relational databases
  - etc.

  are important not only for the direct representation of image information, but also a basis of more complex hierarchical methods of image representation.

Traditional Image Data Structures

- Traditional image data structures, such as
  - matrices
  - chains
  - graphs
  - lists of object properties
  - relational databases
  - etc.

  are important not only for the direct representation of image information, but also a basis of more complex hierarchical methods of image representation.

Traditional Image Data Structures

- Traditional image data structures, such as
  - matrices
  - chains
  - graphs
  - lists of object properties
  - relational databases
  - etc.

  are important not only for the direct representation of image information, but also a basis of more complex hierarchical methods of image representation.

Outline

Matrices

- This is the most common data structure for low level image representation.
- Elements of the matrix are integer, real or complex numbers, corresponding to brightness, or to another property of the corresponding pixel of the sampling grid.
- Image data of this kind are usually the direct output of the image capturing device, e.g., a scanner.
- Pixels of both rectangular and hexagonal sampling grids (Fig. **??**) can be represented by a matrix.
- The correspondence between data and matrix elements is obvious for a rectangular grid; with a hexagonal grid every row in the image is shifted half a pixel to the right or left.

Matrices

- This is the most common data structure for low level image representation.
- Elements of the matrix are integer, real or complex numbers, corresponding to brightness, or to another property of the corresponding pixel of the sampling grid.
- Image data of this kind are usually the direct output of the image capturing device, e.g., a scanner.
- Pixels of both rectangular and hexagonal sampling grids (Fig. **??**) can be represented by a matrix.
- The correspondence between data and matrix elements is obvious for a rectangular grid; with a hexagonal grid every row in the image is shifted half a pixel to the right or left.

Matrices

- This is the most common data structure for low level image representation.
- Elements of the matrix are integer, real or complex numbers, corresponding to brightness, or to another property of the corresponding pixel of the sampling grid.
- Image data of this kind are usually the direct output of the image capturing device, e.g., a scanner.
- Pixels of both rectangular and hexagonal sampling grids (Fig. **??**) can be represented by a matrix.
- The correspondence between data and matrix elements is obvious for a rectangular grid; with a hexagonal grid every row in the image is shifted half a pixel to the right or left.

Matrices

- This is the most common data structure for low level image representation.
- Elements of the matrix are integer, real or complex numbers, corresponding to brightness, or to another property of the corresponding pixel of the sampling grid.
- Image data of this kind are usually the direct output of the image capturing device, e.g., a scanner.
- Pixels of both rectangular and hexagonal sampling grids (Fig. **??**) can be represented by a matrix.
- The correspondence between data and matrix elements is obvious for a rectangular grid; with a hexagonal grid every row in the image is shifted half a pixel to the right or left.

Matrices

- This is the most common data structure for low level image representation.
- Elements of the matrix are integer, real or complex numbers, corresponding to brightness, or to another property of the corresponding pixel of the sampling grid.
- Image data of this kind are usually the direct output of the image capturing device, e.g., a scanner.
- Pixels of both rectangular and hexagonal sampling grids (Fig. **??**) can be represented by a matrix.
- The correspondence between data and matrix elements is obvious for a rectangular grid; with a hexagonal grid every row in the image is shifted half a pixel to the right or left.

Images Represented by Matrices

- Some special images that are represented by matrices are:
  - Binary images (an image with two brightness levels only) is represented by a matrix containing zeros and ones.
  - Several matrices can contain information about one multi-spectral image. Each of these matrices contains one image corresponding to one spectral band.
  - Matrices of different resolution are used to obtain hierarchical image data structures.
  - A representation of a segmented image by a matrix usually saves memory than an explicit list of all spatial relations between all objects, although sometimes we need to record other relations among objects.

Images Represented by Matrices

- Some special images that are represented by matrices are:
  - Binary images (an image with two brightness levels only) is represented by a matrix containing zeros and ones.
  - Several matrices can contain information about one multi-spectral image. Each of these matrices contains one image corresponding to one spectral band.
  - Matrices of different resolution are used to obtain hierarchical image data structures.
  - A representation of a segmented image by a matrix usually saves memory than an explicit list of all spatial relations between all objects, although sometimes we need to record other relations among objects.

Images Represented by Matrices

- Some special images that are represented by matrices are:
  - Binary images (an image with two brightness levels only) is represented by a matrix containing zeros and ones.
  - Several matrices can contain information about one multi-spectral image. Each of these matrices contains one image corresponding to one spectral band.
  - Matrices of different resolution are used to obtain hierarchical image data structures.
  - A representation of a segmented image by a matrix usually saves memory than an explicit list of all spatial relations between all objects, although sometimes we need to record other relations among objects.

Images Represented by Matrices

- Some special images that are represented by matrices are:
  - Binary images (an image with two brightness levels only) is represented by a matrix containing zeros and ones.
  - Several matrices can contain information about one multi-spectral image. Each of these matrices contains one image corresponding to one spectral band.
  - Matrices of different resolution are used to obtain hierarchical image data structures.
  - A representation of a segmented image by a matrix usually saves memory than an explicit list of all spatial relations between all objects, although sometimes we need to record other relations among objects.

Images Represented by Matrices

- Some special images that are represented by matrices are:
  - Binary images (an image with two brightness levels only) is represented by a matrix containing zeros and ones.
  - Several matrices can contain information about one multi-spectral image. Each of these matrices contains one image corresponding to one spectral band.
  - Matrices of different resolution are used to obtain hierarchical image data structures.
  - A representation of a segmented image by a matrix usually saves memory than an explicit list of all spatial relations between all objects, although sometimes we need to record other relations among objects.

Images = Matrices = Arrays

- Image information in the matrix is accessible through the co-ordinates of a pixel that correspond with row and column indexes.
- Most programming language use a standard array data structure to represent a matrix.

Images = Matrices = Arrays

- Image information in the matrix is accessible through the co-ordinates of a pixel that correspond with row and column indexes.
- Most programming language use a standard array data structure to represent a matrix.

Images = Matrices = Arrays in C

- In C, an image of 256 gray-levels with dimension $M \times N$ can be stored in a two dimensional array

unsigned char image [M][N]

- The above stored array may be dis-continuous in memory.
- Then another way to represent the image is by a one-dimensional array, which is continuous in the memory

unsigned char image [M * N]

- We need to know the image dimension M and N usually.

Images = Matrices = Arrays in C

- In C, an image of 256 gray-levels with dimension $M \times N$ can be stored in a two dimensional array

unsigned char image [M][N]

- The above stored array may be dis-continuous in memory.
- Then another way to represent the image is by a one-dimensional array, which is continuous in the memory

unsigned char image [M * N]

- We need to know the image dimension M and N usually.

Images = Matrices = Arrays in C

- In C, an image of 256 gray-levels with dimension $M \times N$ can be stored in a two dimensional array

unsigned char image [M][N]

- The above stored array may be dis-continuous in memory.
- Then another way to represent the image is by a one-dimensional array, which is continuous in the memory

unsigned char image [M * N]

- We need to know the image dimension *M* and *N* usually.

Images = Matrices = Arrays in C

- In C, an image of 256 gray-levels with dimension $M \times N$ can be stored in a two dimensional array

  unsigned char image [M][N]

- The above stored array may be dis-continuous in memory.
- Then another way to represent the image is by a one-dimensional array, which is continuous in the memory

  unsigned char image [M * N]

- We need to know the image dimension M and N usually.

Images = Matrices = Structure in C (I)

- Another method is to represent an image by a structure type,

```
typedef struct _image
{
    int rows;
    int columns;
    char *values;
    <other useful information>;
} Image;
```

or use a pointer to the above structure

```
typedef Image * ImageX;
```

or declared as

```
typedef struct _image * ImageX;
```

Images = Matrices = Structure in C (II)

- When initialized,

  value = (char *) malloc (sizeof(char) * rows * columns);

- To access one element of the matrix at pixel (m, n), use the following macro

```
#define Pixel(IX, m, n)
        (IX->values[m*(IX->columns)+n])
```

where $0 \leq m \leq M - 1$ and $0 \leq n \leq N - 1$.

Images = Matrices = Structure in C (II)

- When initialized,

  value = (char *) malloc (sizeof(char) * rows * columns);

- To access one element of the matrix at pixel (m, n), use the following macro

```
#define Pixel(IX, m, n)
                    (IX->values[m*(IX->columns)+n])
```

where $0 \leq m \leq M - 1$ and $0 \leq n \leq N - 1$.

Other Image Data Types

- You may use other type such as

  float *values;

  or

  double *values;

  to implement much precise computation.

Outline

- Levels of Representation
- Traditional Image Data Structures
  - Matrices
  - Chains
  - Topological Data Structures
  - Relational Structures
- Hierarchical Data Structures
  - Pyramids
  - Quadtrees

Chains

- Chains are used for the representation of object borders in computer vision.
- One element of the chain is a basic symbol, which corresponds to some kind of primitives in the image.
- Chains are appropriate for data that can be arranged as a sequence of symbols.
- The neighboring symbols in a chain usually correspond to the neighboring of primitives in the image.

Chains

- Chains are used for the representation of object borders in computer vision.
- One element of the chain is a basic symbol, which corresponds to some kind of primitives in the image.
- Chains are appropriate for data that can be arranged as a sequence of symbols.
- The neighboring symbols in a chain usually correspond to the neighboring of primitives in the image.

Chains

- Chains are used for the representation of object borders in computer vision.
- One element of the chain is a basic symbol, which corresponds to some kind of primitives in the image.
- Chains are appropriate for data that can be arranged as a sequence of symbols.
- The neighboring symbols in a chain usually correspond to the neighboring of primitives in the image.

Chains

- Chains are used for the representation of object borders in computer vision.
- One element of the chain is a basic symbol, which corresponds to some kind of primitives in the image.
- Chains are appropriate for data that can be arranged as a sequence of symbols.
- The neighboring symbols in a chain usually correspond to the neighboring of primitives in the image.

Chain Codes

- Chain codes or Freeman codes [Freeman, 1961] are often used for the description of object borders, or other one-pixel-wide primitives in images.
- Chain codes describe an object by a sequence of unit-size length line segments with a given orientation.
- The border is defined by the co-ordinates of its reference pixel and the sequence of symbols for lines of unit length in several predefined orientations.
- The first element of such a sequence must bear information about its position to permit the region to be reconstructed.
- The result is a sequence of numbers, indicating the orientation.
- A chain code is of relative nature; data are expressed with respect to some reference point.

Chain Codes

- `Chain codes` or `Freeman codes` [Freeman, 1961] are often used for the description of object borders, or other one-pixel-wide primitives in images.
- Chain codes describe an object by a sequence of unit-size length line segments with a given orientation.
- The border is defined by the co-ordinates of its reference pixel and the sequence of symbols for lines of unit length in several predefined orientations.
- The first element of such a sequence must bear information about its position to permit the region to be reconstructed.
- The result is a sequence of numbers, indicating the orientation.
- A chain code is of relative nature; data are expressed with respect to some reference point.

Chain Codes

- `Chain codes` or `Freeman codes` [Freeman, 1961] are often used for the description of object borders, or other one-pixel-wide primitives in images.
- Chain codes describe an object by a sequence of unit-size length line segments with a given orientation.
- The border is defined by the co-ordinates of its reference pixel and the sequence of symbols for lines of unit length in several predefined orientations.
- The first element of such a sequence must bear information about its position to permit the region to be reconstructed.
- The result is a sequence of numbers, indicating the orientation.
- A chain code is of relative nature; data are expressed with respect to some reference point.

Chain Codes

- `Chain codes` or `Freeman codes` [Freeman, 1961] are often used for the description of object borders, or other one-pixel-wide primitives in images.
- Chain codes describe an object by a sequence of unit-size length line segments with a given orientation.
- The border is defined by the co-ordinates of its reference pixel and the sequence of symbols for lines of unit length in several predefined orientations.
- The first element of such a sequence must bear information about its position to permit the region to be reconstructed.
- The result is a sequence of numbers, indicating the orientation.
- A chain code is of relative nature; data are expressed with respect to some reference point.

Chain Codes

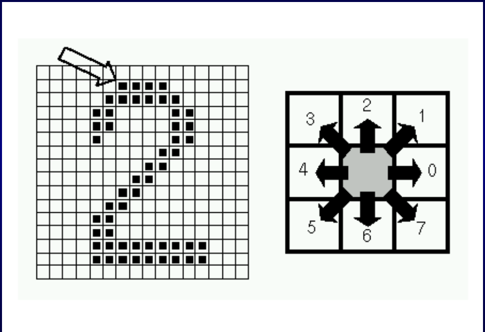
- `Chain codes` or `Freeman codes` [Freeman, 1961] are often used for the description of object borders, or other one-pixel-wide primitives in images.
- Chain codes describe an object by a sequence of unit-size length line segments with a given orientation.
- The border is defined by the co-ordinates of its reference pixel and the sequence of symbols for lines of unit length in several predefined orientations.
- The first element of such a sequence must bear information about its position to permit the region to be reconstructed.
- The result is a sequence of numbers, indicating the orientation.
- A chain code is of relative nature; data are expressed with respect to some reference point.

Chain Codes

- `Chain codes` or `Freeman codes` [Freeman, 1961] are often used for the description of object borders, or other one-pixel-wide primitives in images.
- Chain codes describe an object by a sequence of unit-size length line segments with a given orientation.
- The border is defined by the co-ordinates of its reference pixel and the sequence of symbols for lines of unit length in several predefined orientations.
- The first element of such a sequence must bear information about its position to permit the region to be reconstructed.
- The result is a sequence of numbers, indicating the orientation.
- A chain code is of relative nature; data are expressed with respect to some reference point.

Chain Codes: Example

- An example of a chain code is shown in the following figure, where 8-neighborhoods are used — it is possible to define chain code using 4-neighborhoods as well.



The reference pixel is marked by an arrow. The chain code is

00077665555556600000006444444442221111112234445652211

Chain Codes: Discussions

- To exploit the position invariance of chain codes, e.g., when used for matching, the first element, which contains the position information, should be omitted. One need a method to normalize the chain codes.
- A chain code is very sensitive to noise and arbitrary changes in scale and rotation may cause problems if used fro recognition.
- The description of an image by chains is appropriate for syntactic pattern recognition that is based on formal language theory approaches.
- Chains can be represented using static data structures (e.g., 1D arrays); their size is the longest length of the chain expected.
- Dynamic data structures are more advantageous to save memory.

Chain Codes: Discussions

- To exploit the position invariance of chain codes, e.g., when used for matching, the first element, which contains the position information, should be omitted. One need a method to normalize the chain codes.
- A chain code is very sensitive to noise and arbitrary changes in scale and rotation may cause problems if used fro recognition.
- The description of an image by chains is appropriate for syntactic pattern recognition that is based on formal language theory approaches.
- Chains can be represented using static data structures (e.g., 1D arrays); their size is the longest length of the chain expected.
- Dynamic data structures are more advantageous to save memory.

Chain Codes: Discussions

- To exploit the position invariance of chain codes, e.g., when used for matching, the first element, which contains the position information, should be omitted. One need a method to normalize the chain codes.
- A chain code is very sensitive to noise and arbitrary changes in scale and rotation may cause problems if used fro recognition.
- The description of an image by chains is appropriate for syntactic pattern recognition that is based on formal language theory approaches.
- Chains can be represented using static data structures (e.g., 1D arrays); their size is the longest length of the chain expected.
- Dynamic data structures are more advantageous to save memory.

Chain Codes: Discussions

- To exploit the position invariance of chain codes, e.g., when used for matching, the first element, which contains the position information, should be omitted. One need a method to normalize the chain codes.
- A chain code is very sensitive to noise and arbitrary changes in scale and rotation may cause problems if used fro recognition.
- The description of an image by chains is appropriate for syntactic pattern recognition that is based on formal language theory approaches.
- Chains can be represented using static data structures (e.g., 1D arrays); their size is the longest length of the chain expected.
- Dynamic data structures are more advantageous to save memory.

Chain Codes: Discussions

- To exploit the position invariance of chain codes, e.g., when used for matching, the first element, which contains the position information, should be omitted. One need a method to normalize the chain codes.
- A chain code is very sensitive to noise and arbitrary changes in scale and rotation may cause problems if used fro recognition.
- The description of an image by chains is appropriate for syntactic pattern recognition that is based on formal language theory approaches.
- Chains can be represented using static data structures (e.g., 1D arrays); their size is the longest length of the chain expected.
- Dynamic data structures are more advantageous to save memory.

Outline

Topological Data Structures by Graphs

- Topological data structures describe the image as a set of elements and their relations.
- These relations are often represented using graphs.
- A graph $G = (V, E)$ is an algebraic structure which consists of a set of nodes

$$V = \{v_1, v_2, \cdots, v_n\}, \tag{1}$$

and a set of arcs

$$E = \{e_1, e_2, \cdots, e_m\}. \tag{2}$$

- Each arc e_k is naturally connected with an unordered pair of nodes $\{v_i, v_j\}$ which are not necessarily distinct.
- The degree of the node is equal to the number of incident arcs of the node.

Topological Data Structures by Graphs

- Topological data structures describe the image as a set of elements and their relations.
- These relations are often represented using graphs.
- A graph $G = (V, E)$ is an algebraic structure which consists of a set of nodes

$$V = \{v_1, v_2, \cdots, v_n\}, \tag{1}$$

and a set of arcs

$$E = \{e_1, e_2, \cdots, e_m\}. \tag{2}$$

- Each arc e_k is naturally connected with an unordered pair of nodes $\{v_i, v_j\}$ which are not necessarily distinct.
- The degree of the node is equal to the number of incident arcs of the node.

Topological Data Structures by Graphs

- Topological data structures describe the image as a set of elements and their relations.
- These relations are often represented using graphs.
- A graph $G = (V, E)$ is an algebraic structure which consists of a set of nodes

$$V = \{v_1, v_2, \cdots, v_n\}, \tag{1}$$

and a set of arcs

$$E = \{e_1, e_2, \cdots, e_m\}. \tag{2}$$

- Each arc e_k is naturally connected with an unordered pair of nodes $\{v_i, v_j\}$ which are not necessarily distinct.
- The degree of the node is equal to the number of incident arcs of the node.

Topological Data Structures by Graphs

- Topological data structures describe the image as a set of elements and their relations.
- These relations are often represented using graphs.
- A graph $G = (V, E)$ is an algebraic structure which consists of a set of nodes

$$V = \{v_1, v_2, \cdots, v_n\}, \quad (1)$$

and a set of arcs

$$E = \{e_1, e_2, \cdots, e_m\}. \quad (2)$$

- Each arc e_k is naturally connected with an unordered pair of nodes $\{v_i, v_j\}$ which are not necessarily distinct.
- The degree of the node is equal to the number of incident arcs of the node.

Topological Data Structures by Graphs

- Topological data structures describe the image as a set of elements and their relations.
- These relations are often represented using graphs.
- A graph $G = (V, E)$ is an algebraic structure which consists of a set of nodes

$$V = \{v_1, v_2, \cdots, v_n\}, \tag{1}$$

and a set of arcs

$$E = \{e_1, e_2, \cdots, e_m\}. \tag{2}$$

- Each arc e_k is naturally connected with an unordered pair of nodes $\{v_i, v_j\}$ which are not necessarily distinct.
- The degree of the node is equal to the number of incident arcs of the node.

Evaluated Graphs

- An evaluated graph is a graph in which values are assigned to arcs, to nodes or to both — these values may, e.g., represent weights, or costs.

Region Adjacency Graphs

- The `region adjacency graph` is typical of this class of data structures.
- Nodes correspond to regions and neighboring regions are connected by an arc.
- The segmented image consists of regions of pixels with similar properties (brightness, texture, color, ...) that correspond to some entities in the scene.
- The `neighborhood relation` is fulfilled when the regions have some common border.

Region Adjacency Graphs

- The `region adjacency graph` is typical of this class of data structures.
- Nodes correspond to regions and neighboring regions are connected by an arc.
- The segmented image consists of regions of pixels with similar properties (brightness, texture, color, ...) that correspond to some entities in the scene.
- The `neighborhood relation` is fulfilled when the regions have some common border.

Region Adjacency Graphs

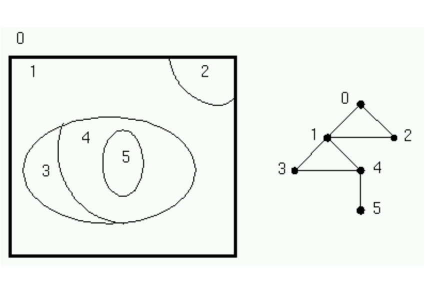
- The `region adjacency graph` is typical of this class of data structures.
- Nodes correspond to regions and neighboring regions are connected by an arc.
- The segmented image consists of regions of pixels with similar properties (brightness, texture, color, ...) that correspond to some entities in the scene.
- The `neighborhood relation` is fulfilled when the regions have some common border.

Region Adjacency Graphs

- The `region adjacency graph` is typical of this class of data structures.
- Nodes correspond to regions and neighboring regions are connected by an arc.
- The segmented image consists of regions of pixels with similar properties (brightness, texture, color, ...) that correspond to some entities in the scene.
- The `neighborhood relation` is fulfilled when the regions have some common border.

Region Adjacency Graphs: Example

An example of an image with regions labeled by numbers and corresponding region adjacency graph is shown in the following figure



The label 0 is denotes pixels out of the image. This value is used to indicate regions that touch borders of the image in the region adjacency graph.

Region Adjacency Graphs: Discussions

- The region adjacency graph has several attractive features.
- If a region enclose other regions, then the part of the graph corresponding with the areas inside can be separated by a cut in the graph.
- Nodes of degree 1 represent simple holes.
- The region adjacency graph can be used for matching with a stored pattern for recognition purpose.

Region Adjacency Graphs: Discussions

- The region adjacency graph has several attractive features.
- If a region enclose other regions, then the part of the graph corresponding with the areas inside can be separated by a cut in the graph.
- Nodes of degree 1 represent simple holes.
- The region adjacency graph can be used for matching with a stored pattern for recognition purpose.

Region Adjacency Graphs: Discussions

- The region adjacency graph has several attractive features.
- If a region enclose other regions, then the part of the graph corresponding with the areas inside can be separated by a cut in the graph.
- Nodes of degree 1 represent simple holes.
- The region adjacency graph can be used for matching with a stored pattern for recognition purpose.

Region Adjacency Graphs: Discussions

- The region adjacency graph has several attractive features.
- If a region enclose other regions, then the part of the graph corresponding with the areas inside can be separated by a cut in the graph.
- Nodes of degree 1 represent simple holes.
- The region adjacency graph can be used for matching with a stored pattern for recognition purpose.

Region Adjacency Graphs: Creation

- The region adjacency graph is usually created from the `region map`, which is a matrix of the same dimension as the original image matrix whose elements are identification labels of the regions.
- To created the region adjacency graph, borders of all regions in the image are traced, and labels of all neighboring regions are stored.
- The region adjacency graph can easily be created from an image represented by a `quadtree` in § 47 as well.

Region Adjacency Graphs: Creation

- The region adjacency graph is usually created from the `region map`, which is a matrix of the same dimension as the original image matrix whose elements are identification labels of the regions.
- To created the region adjacency graph, borders of all regions in the image are traced, and labels of all neighboring regions are stored.
- The region adjacency graph can easily be created from an image represented by a `quadtree` in § 47 as well.

Region Adjacency Graphs: Creation

- The region adjacency graph is usually created from the `region map`, which is a matrix of the same dimension as the original image matrix whose elements are identification labels of the regions.
- To created the region adjacency graph, borders of all regions in the image are traced, and labels of all neighboring regions are stored.
- The region adjacency graph can easily be created from an image represented by a `quadtree` in § 47 as well.

Outline

Relational Structures

- Relational databases can also be used for representation of information from an image.
- All the information is then concentrated in relations between semantically important parts of the image — objects — that are the result of segmentation.
- Relations are recorded in the form of tables.
- Individual objects are associated with their names and other features, e.g., the top-left pixel of the corresponding region in the image.
- Relations between objects are expressed in the relational table as well.

Relational Structures

- Relational databases can also be used for representation of information from an image.
- All the information is then concentrated in relations between semantically important parts of the image — objects — that are the result of segmentation.
- Relations are recorded in the form of tables.
- Individual objects are associated with their names and other features, e.g., the top-left pixel of the corresponding region in the image.
- Relations between objects are expressed in the relational table as well.

Relational Structures

- Relational databases can also be used for representation of information from an image.
- All the information is then concentrated in relations between semantically important parts of the image — objects — that are the result of segmentation.
- Relations are recorded in the form of tables.
- Individual objects are associated with their names and other features, e.g., the top-left pixel of the corresponding region in the image.
- Relations between objects are expressed in the relational table as well.

Relational Structures

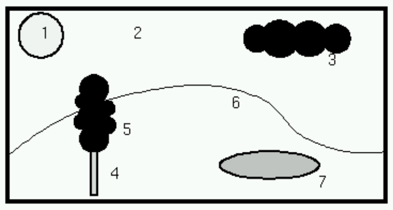
- Relational databases can also be used for representation of information from an image.
- All the information is then concentrated in relations between semantically important parts of the image — objects — that are the result of segmentation.
- Relations are recorded in the form of tables.
- Individual objects are associated with their names and other features, e.g., the top-left pixel of the corresponding region in the image.
- Relations between objects are expressed in the relational table as well.

Relational Structures

- Relational databases can also be used for representation of information from an image.
- All the information is then concentrated in relations between semantically important parts of the image — objects — that are the result of segmentation.
- Relations are recorded in the form of tables.
- Individual objects are associated with their names and other features, e.g., the top-left pixel of the corresponding region in the image.
- Relations between objects are expressed in the relational table as well.

Relational Structures: Example

An example of such a representation is shown in the following figure and table.



(a) Image

No.	Object name	Color	Min. row	Min. col.	Inside
1	sun	white	5	40	2
2	sky	blue	0	0	-
3	cloud	gray	20	180	2
4	tree trunk	brown	95	75	6
5	tree crown	green	53	63	-
6	hill	light green	97	0	-
7	pond	blue	100	160	6

(b) Database

Relational Structures: Discussions

- Description by means of relational structures is appropriate for higher level image understanding.
- Search using keys, similar to database searches, can be used to speed up the whole process.

Relational Structures: Discussions

- Description by means of relational structures is appropriate for higher level image understanding.
- Search using keys, similar to database searches, can be used to speed up the whole process.

Huge Dataset

- Computer vision is by its nature very computationally expensive, if for no other reason than the large amount of data to be processed.
- Systems which we might call sophisticated must process considerable quantities of image data — hundreds of kilobytes to tens of megabytes.
- The visual information perceived by the two human eyes is about 3000 MB/s (add reference or `estimate it`!!!).
- One of the solutions is using parallel computers (in other words brute force).
- Unfortunately many computer vision problems are difficult to divide among processors, or decompose in any way.

Huge Dataset

- Computer vision is by its nature very computationally expensive, if for no other reason than the large amount of data to be processed.
- Systems which we might call sophisticated must process considerable quantities of image data — hundreds of kilobytes to tens of megabytes.
- The visual information perceived by the two human eyes is about 3000 MB/s (add reference or estimate it!!!).
- One of the solutions is using parallel computers (in other words brute force).
- Unfortunately many computer vision problems are difficult to divide among processors, or decompose in any way.

Huge Dataset

- Computer vision is by its nature very computationally expensive, if for no other reason than the large amount of data to be processed.
- Systems which we might call sophisticated must process considerable quantities of image data — hundreds of kilobytes to tens of megabytes.
- The visual information perceived by the two human eyes is about 3000 MB/s (add reference or `estimate it`!!!).
- One of the solutions is using parallel computers (in other words brute force).
- Unfortunately many computer vision problems are difficult to divide among processors, or decompose in any way.

Huge Dataset

- Computer vision is by its nature very computationally expensive, if for no other reason than the large amount of data to be processed.
- Systems which we might call sophisticated must process considerable quantities of image data — hundreds of kilobytes to tens of megabytes.
- The visual information perceived by the two human eyes is about 3000 MB/s (add reference or `estimate it`!!!).
- One of the solutions is using parallel computers (in other words brute force).
- Unfortunately many computer vision problems are difficult to divide among processors, or decompose in any way.

Huge Dataset

- Computer vision is by its nature very computationally expensive, if for no other reason than the large amount of data to be processed.
- Systems which we might call sophisticated must process considerable quantities of image data — hundreds of kilobytes to tens of megabytes.
- The visual information perceived by the two human eyes is about 3000 MB/s (add reference or `estimate it`!!!).
- One of the solutions is using parallel computers (in other words brute force).
- Unfortunately many computer vision problems are difficult to divide among processors, or decompose in any way.

Hierarchical Data Structures

- Hierarchical data structures make it possible to use algorithms which decide a strategy for processing on the basis of relatively small quantities of data.
- They work at the finest resolution only with those parts of the image for which it is necessary, using knowledge instead of brute force to ease and speed up the processing.
- We are going to introduce two typical hierarchical structures, pyramids and quadtrees.

Hierarchical Data Structures

- `Hierarchical data structures` make it possible to use algorithms which decide a strategy for processing on the basis of relatively small quantities of data.
- They work at the finest resolution only with those parts of the image for which it is necessary, using knowledge instead of brute force to ease and speed up the processing.
- We are going to introduce two typical hierarchical structures, pyramids and quadtrees.

Hierarchical Data Structures

- `Hierarchical data structures` make it possible to use algorithms which decide a strategy for processing on the basis of relatively small quantities of data.
- They work at the finest resolution only with those parts of the image for which it is necessary, using knowledge instead of brute force to ease and speed up the processing.
- We are going to introduce two typical hierarchical structures, pyramids and quadtrees.

Outline

Levels of Representation

Traditional Image Data Structures
- Matrices
- Chains
- Topological Data Structures
- Relational Structures

Hierarchical Data Structures
- Pyramids
- Quadtrees

Pyramids

- Pyramids are among the simplest hierarchical data structures.
- We distinguish between `M-pyramids` (matrix pyramids) and `T-pyramids` (tree pyramids).

Pyramids

- Pyramids are among the simplest hierarchical data structures.
- We distinguish between `M-pyramids` (matrix pyramids) and `T-pyramids` (tree pyramids).

M-Pyramids

- A M-pyramid is a sequence $\{M_L, M_{L-1}, \cdots, M_0\}$ of images.
- M_L has the same dimensions and elements as the original image.
- M_{i-1} is derived from the M_i by reducing the resolution, usually, by one half.

M-Pyramids

- A M-pyramid is a sequence $\{M_L, M_{L-1}, \cdots, M_0\}$ of images.
- M_L has the same dimensions and elements as the original image.
- M_{i-1} is derived from the M_i by reducing the resolution, usually, by one half.

M-Pyramids

- A M-pyramid is a sequence $\{M_L, M_{L-1}, \cdots, M_0\}$ of images.
- M_L has the same dimensions and elements as the original image.
- M_{i-1} is derived from the M_i by reducing the resolution, **usually**, by one half.

M-Pyramids: Creation

- When creating pyramids, it is customary to work with square matrices with dimensions equal to powers of 2.
- For images with arbitrary dimensions, a resampling procedure is needed in creating the pyramids.
- M_0 corresponds to one pixel only.
- The number of image pixels used by an M-pyramid for storing all matrices is

$$N^2\left(1+\frac{1}{4}+\frac{1}{16}+\cdots\right)=1.33N^2$$

M-Pyramids: Creation

- When creating pyramids, it is customary to work with square matrices with dimensions equal to powers of 2.
- For images with arbitrary dimensions, a resampling procedure is needed in creating the pyramids.
- M_0 corresponds to one pixel only.
- The number of image pixels used by an M-pyramid for storing all matrices is

$$N^2 \left(1 + \frac{1}{4} + \frac{1}{16} + \cdots\right) = 1.33N^2$$

M-Pyramids: Creation

- When creating pyramids, it is customary to work with square matrices with dimensions equal to powers of 2.
- For images with arbitrary dimensions, a resampling procedure is needed in creating the pyramids.
- M_0 corresponds to one pixel only.
- The number of image pixels used by an M-pyramid for storing all matrices is

$$N^2\left(1+\frac{1}{4}+\frac{1}{16}+\cdots\right)=1.33N^2$$

M-Pyramids: Creation

- When creating pyramids, it is customary to work with square matrices with dimensions equal to powers of 2.
- For images with arbitrary dimensions, a resampling procedure is needed in creating the pyramids.
- M_0 corresponds to one pixel only.
- The number of image pixels used by an M-pyramid for storing all matrices is

$$N^2\left(1+\frac{1}{4}+\frac{1}{16}+\cdots\right)=1.33N^2$$

M-Pyramids: Discussions

- M-pyramid created by shrinking the image dimensions. The matlab script for this example is m_pyramid.m.

M-Pyramids: Discussions

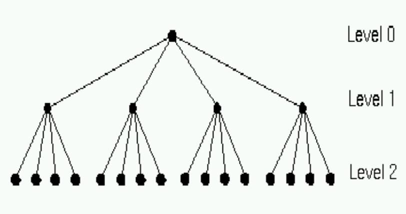
- M-pyramids are used when it is necessary to work with an image at different resolutions simultaneously.
- An image having one degree smaller resolution in a pyramid contains four times less data, so that it is processed approximately four times as quickly.

M-Pyramids: Discussions

- M-pyramids are used when it is necessary to work with an image at different resolutions simultaneously.
- An image having one degree smaller resolution in a pyramid contains four times less data, so that it is processed approximately four times as quickly.

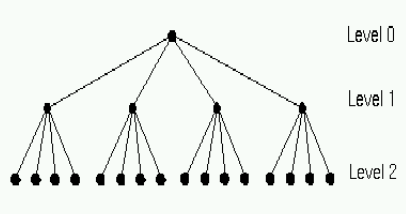
T-Pyramids

- Often it is advantageous to use several images of the same resolution simultaneously rather than to create just one image at a resolution in the M-pyramid.
- E.g., we use images at a resolution, containing additional information at this resolution, texture, orientation and segmentation properties, etc.
- Such images can be represented using tree pyramids — T-pyramids.
- The following figure is a example T-pyramid tree. Every node of the T-pyramid has 4 child nodes.



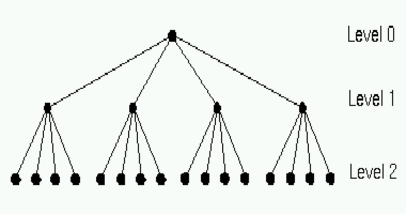
T-Pyramids

- Often it is advantageous to use several images of the same resolution simultaneously rather than to create just one image at a resolution in the M-pyramid.
- E.g., we use images at a resolution, containing additional information at this resolution, texture, orientation and segmentation properties, etc.
- Such images can be represented using tree pyramids — T-pyramids.
- The following figure is a example T-pyramid tree. Every node of the T-pyramid has 4 child nodes.



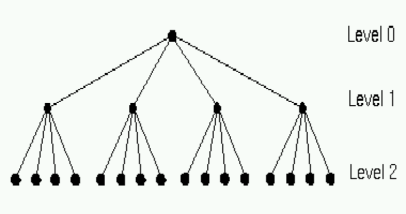
T-Pyramids

- Often it is advantageous to use several images of the same resolution simultaneously rather than to create just one image at a resolution in the M-pyramid.
- E.g., we use images at a resolution, containing additional information at this resolution, texture, orientation and segmentation properties, etc.
- Such images can be represented using tree pyramids — T-pyramids.
- The following figure is a example T-pyramid tree. Every node of the T-pyramid has 4 child nodes.



T-Pyramids

- Often it is advantageous to use several images of the same resolution simultaneously rather than to create just one image at a resolution in the M-pyramid.
- E.g., we use images at a resolution, containing additional information at this resolution, texture, orientation and segmentation properties, etc.
- Such images can be represented using tree pyramids — T-pyramids.
- The following figure is a example T-pyramid tree. Every node of the T-pyramid has 4 child nodes.



Outline

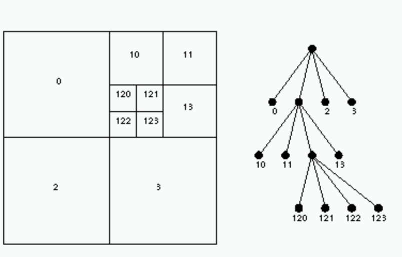
Levels of Representation

Traditional Image Data Structures
- Matrices
- Chains
- Topological Data Structures
- Relational Structures

Hierarchical Data Structures
- Pyramids
- Quadtrees

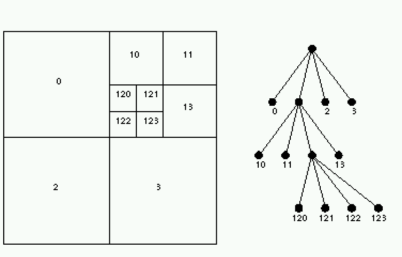
Quadtrees

- Quadtrees are modifications of T-pyramids.
- Every node of the tree except the leaves has four children (NW: north-western, NE: north-eastern, SW: south-western, SE: south-eastern).
- The image is divided into four quadrants at each hierarchical level, though it is not necessary to keep nodes at all levels.
- If a parent node has four children of the same (e.g., brightness) value, (which is often characterized by a similarity measure), it is not necessary to record them.



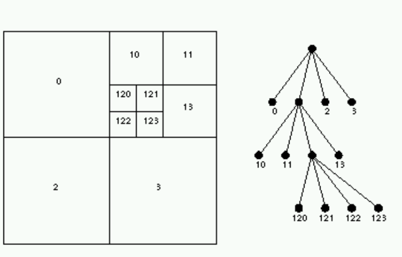
Quadtrees

- Quadtrees are modifications of T-pyramids.
- Every node of the tree except the leaves has four children (NW: north-western, NE: north-eastern, SW: south-western, SE: south-eastern).
- The image is divided into four quadrants at each hierarchical level, though it is not necessary to keep nodes at all levels.
- If a parent node has four children of the same (e.g., brightness) value, (which is often characterized by a similarity measure), it is not necessary to record them.



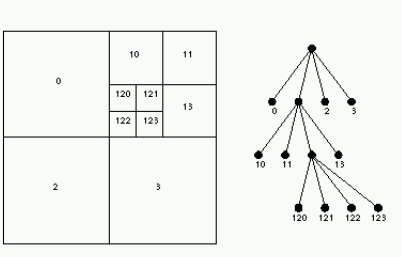
Quadtrees

- Quadtrees are modifications of T-pyramids.
- Every node of the tree except the leaves has four children (NW: north-western, NE: north-eastern, SW: south-western, SE: south-eastern).
- The image is divided into four quadrants at each hierarchical level, though it is not necessary to keep nodes at all levels.
- If a parent node has four children of the same (e.g., brightness) value, (which is often characterized by a similarity measure), it is not necessary to record them.



Quadtrees

- Quadtrees are modifications of T-pyramids.
- Every node of the tree except the leaves has four children (NW: north-western, NE: north-eastern, SW: south-western, SE: south-eastern).
- The image is divided into four quadrants at each hierarchical level, though it is not necessary to keep nodes at all levels.
- If a parent node has four children of the same (e.g., brightness) value, (which is often characterized by a similarity measure), it is not necessary to record them.



Quadtrees: Structures

- Quadtrees are usually represented by recording the whole tree as a list of its individual nodes.
- Every node being a record with several items characterizing it.
- An example is given as following

```
node = {
        node_type,
        pointer_to_NW_son,
        pointer_to_NE_son,
        pointer_to_SW_son,
        pointer_to_SE_son,
        pointer_to_Father,
        other_data
    }
```

- In the item "node_type", there is information about whether the node is a leaf or inside the tree.

Quadtrees: Structures

- Quadtrees are usually represented by recording the whole tree as a list of its individual nodes.
- Every node being a record with several items characterizing it.
- An example is given as following

```
node = {
    node_type,
    pointer_to_NW_son,
    pointer_to_NE_son,
    pointer_to_SW_son,
    pointer_to_SE_son,
    pointer_to_Father,
    other_data
}
```

- In the item "node_type", there is information about whether the node is a leaf or inside the tree.

Quadtrees: Structures

- Quadtrees are usually represented by recording the whole tree as a list of its individual nodes.
- Every node being a record with several items characterizing it.
- An example is given as following

```
node = {
        node_type,
        pointer_to_NW_son,
        pointer_to_NE_son,
        pointer_to_SW_son,
        pointer_to_SE_son,
        pointer_to_Father,
        other_data
    }
```

- In the item “node_type”, there is information about whether the node is a leaf or inside the tree.

Quadtrees: Structures

- Quadtrees are usually represented by recording the whole tree as a list of its individual nodes.
- Every node being a record with several items characterizing it.
- An example is given as following

```
node = {
        node_type,
        pointer_to_NW_son,
        pointer_to_NE_son,
        pointer_to_SW_son,
        pointer_to_SE_son,
        pointer_to_Father,
        other_data
      }
```

- In the item "node_type", there is information about whether the node is a leaf or inside the tree.

Quadtrees: Structures

- Quadtrees are usually represented by recording the whole tree as a list of its individual nodes.
- Every node being a record with several items characterizing it.
- An example is given as following

```
node = {
        node_type,
        pointer_to_NW_son,
        pointer_to_NE_son,
        pointer_to_SW_son,
        pointer_to_SE_son,
        pointer_to_Father,
        other_data
       }
```

- In the item “node_type”, there is information about whether the node is a leaf or inside the tree.

Quadtrees: Structures

- Quadtrees are usually represented by recording the whole tree as a list of its individual nodes.
- Every node being a record with several items characterizing it.
- An example is given as following

```
node = {
        node_type,
        pointer_to_NW_son,
        pointer_to_NE_son,
        pointer_to_SW_son,
        pointer_to_SE_son,
        pointer_to_Father,
        other_data
      }
```

- In the item “node_type”, there is information about whether the node is a leaf or inside the tree.

Quadtrees: Example

- Launch matlab, run the demo "qtdemo".
- matlab uses sparse matrix to store the quadtree decomposition, without the brightness value information for each node or block.

Quadtrees: Example

- Launch matlab, run the demo “qtdemo”.
- matlab uses sparse matrix to store the quadtree decomposition, without the brightness value information for each node or block.

Quadtrees: Discussions

- Disadvantages associated with image pyramid representation:
  - Dependence on the position, orientation and relative size of objects.
  - Two similar images with just very small differences can have very different pyramid or quadtree representations.
  - Even two images depicting the same, slightly shifted scene, can have entirely different representations.
  - These disadvantages can be overcome using a normalized shape of quadtree in which do not create the quadtree for the whole image, but for its individual objects.
  - Please refer the text book for further discussions.

Quadtrees: Discussions

- Disadvantages associated with image pyramid representation:
  - Dependence on the position, orientation and relative size of objects.
  - Two similar images with just very small differences can have very different pyramid or quadtree representations.
  - Even two images depicting the same, slightly shifted scene, can have entirely different representations.
  - These disadvantages can be overcome using a normalized shape of quadtree in which do not create the quadtree for the whole image, but for its individual objects.
  - Please refer the text book for further discussions.

Quadtrees: Discussions

- Disadvantages associated with image pyramid representation:
  - Dependence on the position, orientation and relative size of objects.
  - Two similar images with just very small differences can have very different pyramid or quadtree representations.
  - Even two images depicting the same, slightly shifted scene, can have entirely different representations.
  - These disadvantages can be overcome using a normalized shape of quadtree in which do not create the quadtree for the whole image, but for its individual objects.
  - Please refer the text book for further discussions.

Quadtrees: Discussions

- Disadvantages associated with image pyramid representation:
  - Dependence on the position, orientation and relative size of objects.
  - Two similar images with just very small differences can have very different pyramid or quadtree representations.
  - Even two images depicting the same, slightly shifted scene, can have entirely different representations.
  - These disadvantages can be overcome using a normalized shape of quadtree in which do not create the quadtree for the whole image, but for its individual objects.
  - Please refer the text book for further discussions.

Quadtrees: Discussions

- Disadvantages associated with image pyramid representation:
  - Dependence on the position, orientation and relative size of objects.
  - Two similar images with just very small differences can have very different pyramid or quadtree representations.
  - Even two images depicting the same, slightly shifted scene, can have entirely different representations.
  - These disadvantages can be overcome using a normalized shape of quadtree in which do not create the quadtree for the whole image, but for its individual objects.
  - Please refer the text book for further discussions.

Quadtrees: Discussions

- Disadvantages associated with image pyramid representation:
  - Dependence on the position, orientation and relative size of objects.
  - Two similar images with just very small differences can have very different pyramid or quadtree representations.
  - Even two images depicting the same, slightly shifted scene, can have entirely different representations.
  - These disadvantages can be overcome using a normalized shape of quadtree in which do not create the quadtree for the whole image, but for its individual objects.
  - Please refer the text book for further discussions.

References I

Freeman, H. (1961).
On the envoding of arbitrary geometric confgiuration.
IRE Transactions on Electronic Computers, EC-10(2):260 – 268.