M.Tech. (Computer Science) Dissertation Series

Content Based Image Retrieval using Shape and Color Feature

A dissertation submitted in partial fulfillment of the requirements for the Master of Technology (Computer Science) degree of the Indian Statistical Institute

By

Bibhas Chandra Dhara

Under the supervision of

Prof. Bhabatosh Chanda

ECSU



INDIAN STATISTICAL INSTITUTE

203, Barrackpore Trunk Road

Kolkata – 700 108

2002

CERTIFICATE OF APPROVAL

This is to certify that the thesis entitled "***Content Based Image Retrieval using Shape and Color Feature***" is an authentic record of the dissertation carried out by **Bibhas Chandra Dhara** under my supervision and guidance. The work fulfils the requirements for the award of the M.Tech. degree in Computer Science.

Dated : The 19th of July, 2002.

B. Chanda

Signed :

(Prof. B. Chanda)

Supervisor

P.K. Nandi 19/7/02

Countersigned :

External Examiner

ACKNOWLEDGEMENTS

It has been a great honour and a rewarding experience to work under the auspices of a guide as eminent as Prof. Bhabatosh Chanda, ECSU. Endowed with the amazing capability of making complex mathematical expressions and impossible-to-understand theories seem as simple and interesting as Enid Blyton stories, it has been Prof. Chanda's persistent efforts and encouragement that has borne fruits in the successful completion of this project. No amount of thanks can repay his contribution to this work.

I am fortunate enough to be associated with an Institute of international repute that has provided me excellent infrastructure and environment in helping me complete my project. I take this opportunity to thank all staff and Research Scholars of ECSU whose presence allowed me to work freely at late hours and Saturdays.

Bibhas Chandra Dhara

Bibhas Chandra Dhara

July 19, 2002

Indian Statistical Institute, Kolkata

Contents

1 Introduction **2**

1.1 What is CBIR system? 3

1.2 Interfacing 3

1.3 Available CBIR software 4

1.3.1 Commercial system 4

1.3.2 Experimental systems 4

1.4 Practical applications of CBIR 5

2 Overview of the project **5**

3 Description of proposed system **6**

3.1 Feature Extraction 6

3.1.1 Color Feature 6

3.1.2 Shape Feature 8

3.1.3 Texture Feature 15

3.2 Multidimensional Indexing 16

3.3 Retrieval Engine 17

4 Implementation of User interface **17**

5 Experimental Results **19**

5.1 Performance Analysis 20

6 Conclusion **20**

References **43**

1 Introduction

The term "Content Based Images Retrieval System" has two important parts: Content of images and Image retrieval.

Content of image: An image can be describe by several features, which include shape, color, texture and spatial relationship are called content of image. By using these features, images can be modeled as well as can be indexed in an image database. An image retrieval system which is based on these low level features are called Content Based Images Retrieval (CBIR) System.

Image Retrieval: How to retrieve images similar to a query-image is becoming an important problem with the rapid increase of media information on the web. Generally, users wants to provide query images and use some kind of system which can present them with a set of similar images. Therefore, how to describe and model an image, how to compare different images and judge whether they are similar, how to construct an index of image database, and how to conduct search efficiently are some of the key problems in any so-called image retrieval system.

Interest in the potential of digital images has increased enormously over the last few years, fuelled at least in part by the rapid growth of imaging on the World-Wide Web. Users need to *retrieve images* from a collection come from a variety of domains, including crime prevention (e.g. finger print identification, face recognition), medicine (e.g. case-based diagnosis from radiographs or scan data), education, architecture, fashion, publishing, and so on. This images are stored may be in a database or may be on many databases scattered over various geographical locations. So a huge information is out there and can easily be accessed. Information can not be accessed unless it is properly organized, because searching and locating a desired piece of image from a varied and large collection usually result in a total frustration [1]. Two major research communities, Database Management and Computer Vision, are putting considerable effort towards the solution of this problem. Accordingly two major approaches have emerged: text based and visual based image retrieval.

In *text-based* Database Management System, images are first manually annotated using a set of keywords that describe the content of the image. Actually here images are indexed and arranged using these keywords, finally images are retrieved based on a text-based query. This text-based image retrieval techniques face two major problem: a *labour intensiveness* and *annotation impreciseness.*

CBIR operates on a totally different principle from keyword indexing. Primitive features characterizing image content, such as color, texture, and shape, are computed for both stored and query images, and used to identify (say) the 20 stored images that are most closely matching the query.

1.1 What is CBIR system?

It would appear that Kota was the first person to use the term *content based image retrieval*. He used it describe his experiments about automatic retrieval of images from large image database, by color and shape features. Since then, the term has been widely used to represent the process of retrieving desired images on the basis of the features (such as color, shape and texture) which can be automatically extracted from the images themselves.

CBIR differs from classical information retrieval in that image database are essentially unstructured. One key issue with any kind of image processing is the need to extract useful information from the raw data. Image databases thus differ fundamentally from text databases, where the raw material (word stored as ASCII character string) has already been logically structured by the another. There are basic issues to be consider with CBIR system:

- Understanding image users' needs and information seeking behavior.
- Identification of suitable ways of describing image content.
- Extracting features from raw images.
- Providing compact storage for large image database.
- Matching query and stored images in a way that reflects human similarity judgments.
- Efficiently accessing stored images by content.
- Providing usable human interface to CBIR system.

1.2 Interfacing

A tool that helps the users to express their search needs accurately and easily is crucial in any retrieval system. Images retrieval system is no exception of this. Some system e.g. GRIM-DBMS [2] use SQL-like query language.

Most useful and obvious way to express query is *query-by-example*: providing a simple of the kind of output desired and asking the system to retrieve further examples of the same kind. Virtually all current CBIR systems now offer query-by-example searching, where users submit a query image and the system retrieves and display output images.

Some times it may happen that user have no example in hand. Several alternative query formulation methods have been proposed here, most based on ideas originally developed for IBM's QBIC system [3]. The original QBIC system interface allowed users to specify color queries either by slides varying the relatives amounts of red, green and blue in the

query, or by selecting a desired color from palette. Texture queries could also be specified by choosing from a pallete , and shape by sketching the desired objects on the screen [4].

The ability to refine searches online in response to user indications relevance, known as *relevance feedback*, is particularly useful in image retrieval, even though it was originally developed for text [5]. This is firstly because users can normally judge the relevance of a set of images displayed on the screen within seconds, and secondly because so few current systems are capable of matching users' needs accurately first time round. The usefulness of relevance feedback for image retrieval was first demonstrated within a keyword-based system [6]; the facility has now been successfully implemented in several CBIR systems (e.g. [7] [8]).

1.3 Available CBIR software

Several image retrieval systems are now available for use. These CBIR systems can be grouped into two categories: *Commercial system* and *Experimental system.*

1.3.1 Commercial system

QBIC: Query By Image Content or QBIC system is developed by IBM. It offers retrieval by any combnation of color, texture or shape- as well as by text keywords. Image queries can be formulated by selection from a palette, specifying an example query image, or sketching a desired shape on the screen. The system extracts and stores color,shape and texture features from each image added to the database, and uses R*-tree indexes to improve search efficiency. The demo of this system is available at http://wwwwqbic.almadean.ibm.com/.

Virage: Another well-known commercial system is the VIR Image Engine from Virage. This is available as a series of independent modules, which systems developers can build in to their own programs. An on line demonstration of the VIR Image Engine can be found at http://www.virage.com/online/. A high-profile application of Virage technology is AltaVista's AV Photo Finder (http://image.altavista.com/cgi-bin/avncgi), allowing Web surfers to search for images by content similarity.

1.3.2 Experimental systems

A large number of experimental system have been developed, mainly by academic institutions, in order to demonstrate the feasibility of new techniques. Many of these are available as demonstartion on Web. Some of the best-known are described below.

Photobook:The Photobook was developed at MIT. It is a tool for performing queries on image database, based on image content. Basically, just like the other CBIR systems, it compares the features of images, not the images themselves. These features include color,

shape, and texture. The largest difference between Photobook and other CBIR systems (e.g., QBIC) is that the Photobook system includes an interactive learning agent, called FourEyes [9]. This agent selects and combines models based on instances from users. Its online demo is available at http://www-white.media.mit.edu/vismod/demos/photobook/.

VisualSEEK and WebSEEK:VisualSEEK system is the first of a whole family of experimental systems developed at Columbia University,New York. It offers searching by image region color,shape and spatial relation,as well as keywords. Users can build up image queries by specifying areas of defined shape and color at absolute or relative locations within image. The WebSEEK system aims to facilitate image searching on the Web. Demos are available at http://www.ee.columbia.edu/ sfchang/ demos.html and http://disney.ctr.columbia.edu/WEBSEEK/.

1.4 Practical applications of CBIR

A wide range of possible applications are for CBIR technology has been identified. Potentially fruitful areas include:

- Crime prevention:
  Law enforcement agencies typically maintain large archives of visual evidence, including past suspects' facial photographs,fingerprints,tyre treads and shoeprints. Whenever a serious crime is committed, they can compare evidence from the scene of the crime for its similarity to records in their archives. Strictly speaking, this is an example of *identity* rather than *similarity* matching.

- The military:
  Military applications of imaging technology are probably the best-developed, though least publicized. Recognition of enemy aircraft from radar screens, identification of targets from satellite photograph are known examples. Many of the techniques used in crime prevention could also be relevant to the military field.

- Intellectual property:
  Trademark image registration, where a new candidate mark is compare with existing marks to ensure that there is no risk of confusion, has long been recognized as a prime application area.

2 Overview of the project

Objective of this work is to develop a CBIR system based on shape and color features. As we intend to build up a complete system the associated jobs are

1. Building up a pilot image database on which the system will be tested.

2. Building up a user interface to facilitate image retrieval as well as updating the image database.

3. Try to evaluate quantitatively the performance of the developed system.

To build up an intelligent database system first of all we need to collect a large number of images from variety of daomains and images collected from each domain possess some kind of similarity. Use efficient indexing methods for storing purpose. For detail deserip-tion see later. To retrieve images from database or for updating database there should be a medium through which user can interact with the system. For this purpose GUI is most suitable and it is easier for the user to interact. The interface with browsing and sketching facilites are desirable. For detail see section 1.2 and section 4. When we build a system definitely it is necessary to evaluate the performance of the system. How efficient the system is? This type of question is there when we talk about performance of the system. To evaluate the system performance retrieved results should be compared with the groundtruth data, and then find the percentage of success. For detail see section 5.

3 Description of proposed system

There are three fundamental modules in a CBIR system, namely,

1. Visual Feature Extraction.

2. Multidimensional indexing , and

3. Retrieval Engine.

3.1 Feature Extraction

In CBIR system, visual features are chosen so that they should be able to describe an image. Here objective is to represent the visual pattern by a minimum set of features or properties that characterizes the scene. Features may be classified into two broad categories: *high-level features* and *low-level features*. Low-level features (color, shape, texture) are most important for CBIR system. The features of the object in the image file are extracted when an image file is added to the database. These features are stored in from of a record in file. Similarly when an image file is removed from the collection, the record corresponding to its feature vector is also removed from the file.

3.1.1 Color Feature

Among all features color is probably the most straightforward one which enables human to recognize different images. Color similarity measure becomes a very important aspect

in the implementation of the CBIR system. The main advantage of this feature is its invariant to size, position, orientation and arrangements of the objects. Color histogram is a popular image feature that has been exploited by many image retrieval systems. All color histogram based image retrieval systems are composed of at least the following three operations:

- **Color space subdivision:** Given a color space C, subdivide C into n cells such that C=$\{c_i$:i=1,. . . ,n$\}$
- **Histogram creation:** The histogram of image I in color space C is define as $H_C(I)$=$\{$#(I,c_i):i=1,. . . ,n$\}$ where #(I,c_i) is the number of I's pixel that falls in cell c_i of color space C.
- **Histogram matching:** Given a similarity metric function $\mathcal{S}$, the similarity of a pair of images I,J is defined as $\mathcal{S}(H_C(I),H_C(J))$.

  The above description shows that there are three important factors: The color space C, the subdivision number n, and the similarity function $\mathcal{S}$ that must be determined before we can create, and make use of histogram to retrieve images. Different systems give different choices for each of these factors, which in turn determine the characteristics of the system.

Color feature vector used in system

We know that any color histogram system begins with a partition of color space model. We partition the color system using the HSV (hue-value-saturation) model. Hue value for each pixel lying inside the object boundary is established (since we are concerned about color of the object in the image file). Next step is to create a hue histogram for the image file, we create a histogram of frequency n_i (number of pixels having hue value i, i ranging from 0^0 to 359^0) versus i. But a basic question comes in mind. Are these histogram fit for comparison purposes in this form? Suppose, we have two images, all the pixels in the first image have hue value 5^0 and all the pixel in the second image have hue value 358^0. How do you compare them? Ideally a query on an image in which all the pixels have hue 0^0 should yield the second image as the best match as the angular distance of 358^0 is less than the angular distance of 5^0 from 0^0. In other words the second image is more red than the first, due to the same reasons first image is more yellow than the second one. Therefore, we need to measure the color effect of each pixel on each point in the hue circle. Obviously the effect decreases as the distance between hue and the point on the hue circle increases. Secondly, to reduce computation during search and retrieval we use only a few values of hue instead of sampling the histogram at 360 points.

Let N_i be a measure of the effect of the color of each pixel on i-th point on hue circle (point starts with index 0) then

$$N_i = \sum_{j=-179}^{179} (180 - \mid j \mid) * n_{(i+j) mod 360}$$

We can thus now obtain the histogram of N_i versus i (i=1,. . . ,n). We sample this histogram at the points 0^0, 60^0, 120^0, 180^0, 240^0, 300^0 to obtain N_i values at these points. If hue value is saturated then if value of any RGB component is less than 128 we count this as black point otherwise count as white point. We also use these two as feature component. These eight values will form the feature of our color feature vector and can be used for comparison purposes. These feature components are independent from *rotation, scaling* and *translation.*

3.1.2 Shape Feature

The shape of objects contained in an image is another important feature of the content of the image. In many situation , people can recognize an object only by its shape. Shape may be defined as the description of an object minus its position, orientation and size.

The shape of an object can be obtained by tracking its boundary. Generally, there are two types of approaches which are used in shape analysis: one is based on the region of objects (e.g., Moment invariant) ; other is based on the boundary of objects (e.g., Fourier descriptor). So for shape of an object boundary is very important, for that segmentation is very important to image retrieval and segmentation algorithms are also included in many CBIR system, consequently, result of content based image retrieval largely depends on the performance of segmentation algorithms. However, since segmentation is a psycho-physical problem [10] and no algorithm can segment all kinds of images with equal satisfaction, interactive mathods are found more useful in many cases. Ideally for a fully automated system, we would want a generic segmentation algorithm to do this task for us.

Used Segmentation Method

For the graylevel images segmentation methods are broad classify into categories: Derivative approach, Pattern fitting approach. Here we use Derivative approach. In Derivative approach firstly the graylevel image produce another image,usually called gradient image, by applying derivative operators. Then this gradient image is made to undergo thresholding for segmentation.

Let g(r,c) is input image and gradient image is

$$g'(r,c) = \nabla g(r,c)$$

For simplicity let g(r,c) be written as g_0 and graylevel of its 8-neighbour pixels g(r-1,c), g(r-1,c-1), g(r,c-1), g(r+1,c-1), g(r+1,c), g(r+1,c+1), g(r,c+1),g(r-1,c+1) be written as g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, respectively as shown in Figure 1.

For masking purpose we used four matrices m_1, m_2, m_3, m_4 respectively which are shown in Figure 2

g_2	g_1	g_8
g_3	g_0	g_7
g_4	g_5	g_6

Figure 1: 8-neighbour of g_0

1	$\sqrt{2}$	1
0	0	0
-1	$-\sqrt{2}$	-1

-1	0	1
$-\sqrt{2}$	0	$\sqrt{2}$
-1	0	1

0	1	$\sqrt{2}$
-1	0	1
$-\sqrt{2}$	-1	0

$\sqrt{2}$	1	0
1	0	-1
0	-1	$-\sqrt{2}$

Figure 2: Masking operator m_1, m_2, m_3, m_4

- compute

$$d_i = \frac{(g * m_i)}{3.414} : i = 1, 2, 3, 4$$

where (g*m_i)= convolution of g and m_i
and then consider

$$g'(r, c) = max(d_1, d_2, d_3, d_4)$$

- Thresholding method

1. compute $frequency_i$ for all gradient value i.
2. Let

$$thresholdvalue = \frac{matrix\ size * 15}{100}$$

3. Let t be the largest value s.t.

$$\sum_{j=1}^{t} frequency_j \leq thresholdvalue$$

4. construct binary matrix

$$bin(r, c) = \begin{cases} 0 & \text{if } g'(r, c) \leq \text{t} \\ 1 & \text{otherwise} \end{cases}$$

- Consider four matrices Mathorz,Matvert,Mat45,Mat135 each are initialized with 0 then,

1. get the positions of left most and right most blocks of 0's from *bin* matrix and assign 1 at same positions of Mathorz.

g_2	g_3	g_4
g_1	g_0	*
*	*	*

Figure 3: used for labeling

2. do same for uppermost and bottom most blocks of 0's of *bin* and assign value to Matvert.
3. do same for extremes blocks of 0's of *bin* along slope 45^0 and assign values to Mat45
4. do same for extremes blocks of 0's of *bin* along slope 135^0 and assign values to Mat135

•

$$MAT = (Mathorz\ OR\ Matvert) AND (Mat45\ OR\ Mat135)$$

- Complement the matrix MAT to get object regions.
- Component labeling

  Component labeling is performed on the matrix MAT considering 8-neighbours. Consider the matrix in Figutre 3.

  1. if($g_0 = 1$) then
     if(g_i=1) for any value of i,1 $\leq$ i $\leq$ 4 then
     g_0 and those g_i's have same label.

  2. if($g_0 = 1$) AND g_i=0 for all i,1 $\leq$ i $\leq$ 4 then
     new label will be assigned to g_0.

- Find largest component by pixel counting.

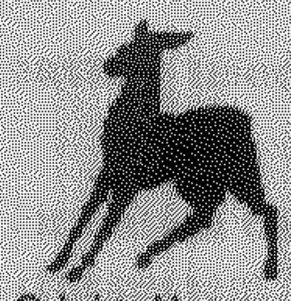
The result is a binary image called the *Objectmap*.

Edgemap: By applying this segmentation algorithm we genetare *Edgemap(i,j)* for the object associated with the image file.

$$Edgemap(i,j) = \begin{cases} 1 & \text{if (i,j) pixel is a boundary pixel} \\ 0 & \text{otherwise} \end{cases}$$

So we get the object in the image file. Some results of the segmentation algorithm are shown in Figure 4.

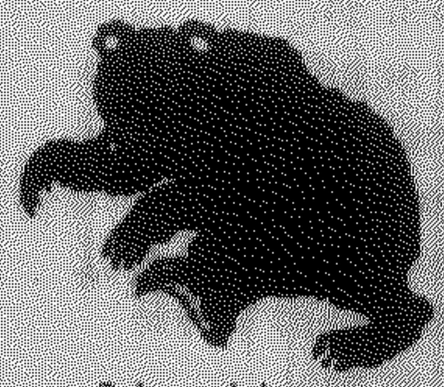
The object which is identified by applying segmentation algorithm may not always occupy whole portion of image file. So to reduce time complexity for scan the object we

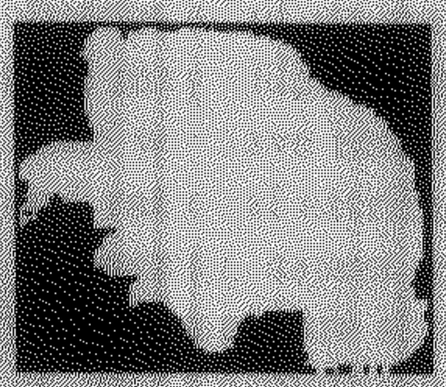

Original Image

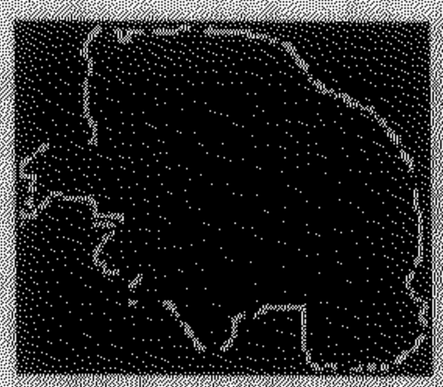

Objectmap of the Image

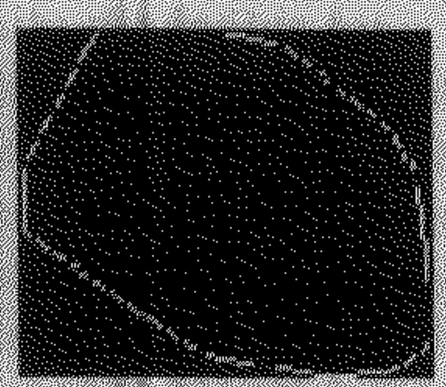

Edgemap of the Image

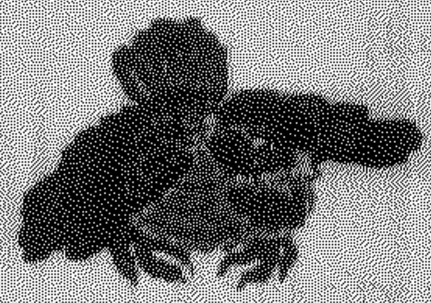

ConvexHull of Edgemap

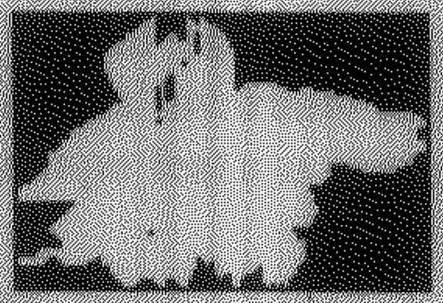

Original Image

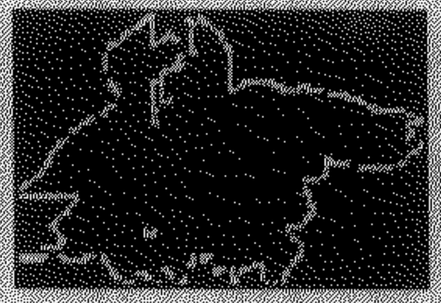

Objectmap of the Image

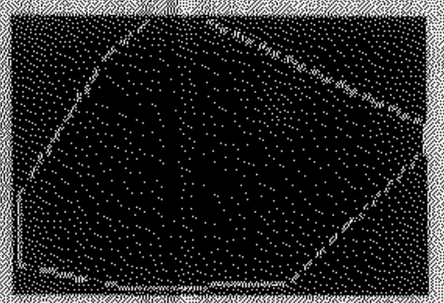

Edgemap of the Image

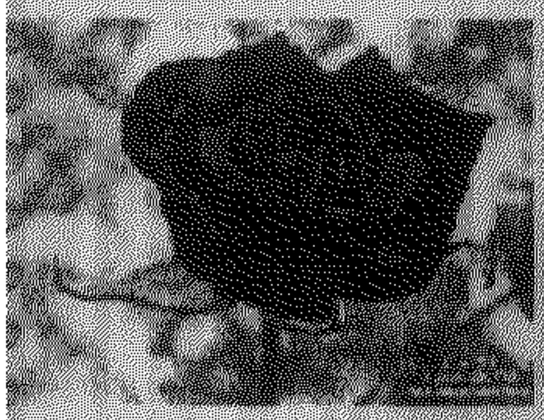

ConvexHull of Edgemap

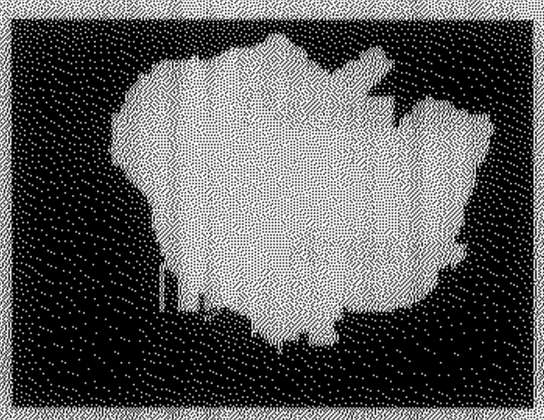

Original Image

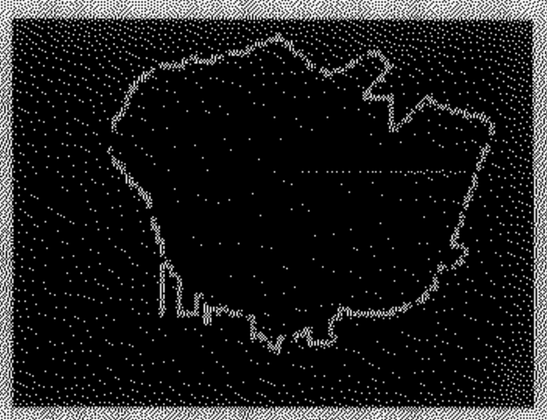

Objectmap of the Image

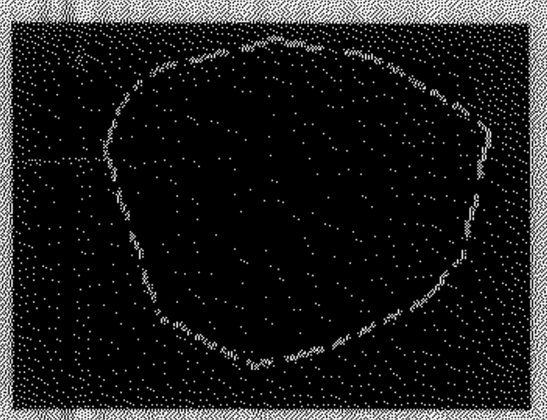

Edgemap of the Image

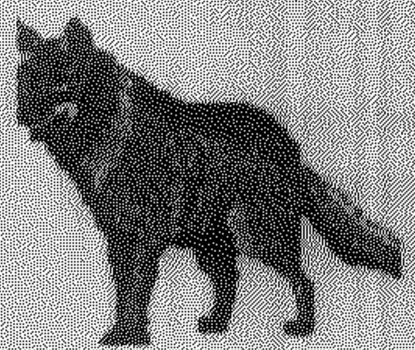

ConvexHull of Edgemap

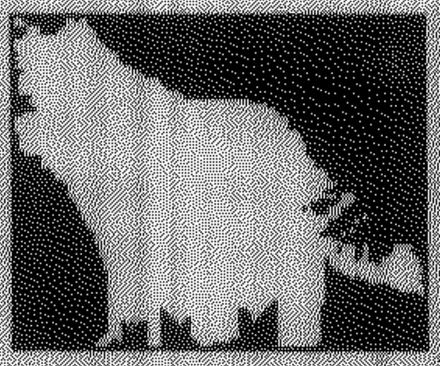

Original Image

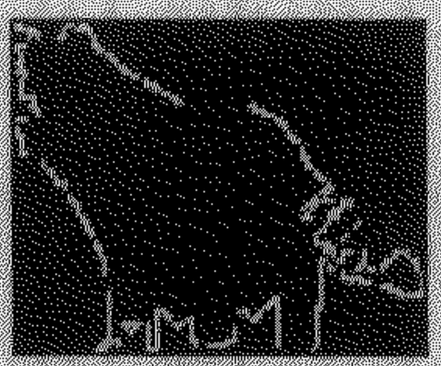

Objectmap of the Image


Edgemap of the Image


ConvexHull of Edgemap


Original Image


Objectmap of the Image


Edgemap of the Image

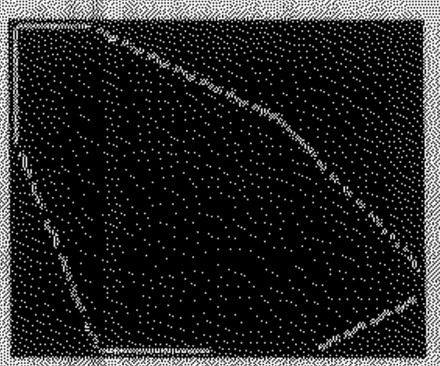

ConvexHull of Edgemap

Figure 4: Output of Segmentation and Convex approximation

find out smallest covering rectangle (this is an implementation issue).

Shape feature used

Major information content of an image is the shape of the objects present in it and their relative arrangement. Shape can roughly be defined as the description of an object minus its position, orientation and size. Therefore, shape features used for object identification or mathing should be invariant to *translation, rotation,* and *scale.* Shape features are those properties of objects or of its parts that are same for visually dissimilar type of objects. Shape features may calculated for the whole image or for each individual objects. In the former case we have mainly topological attributes like, *number of objects, Euler number,* etc. and rational attributes(among objects). In the latter case features like *area, perimeter, convexity, aspect ratio, circularity* and *elongateness* are computed. Most of the CBIR systems measure shape features either by *geometric moments* or by *Fourier descriptor* [11, 12, 13] methods. Algorithms for generating and searching for a given geometric invariant is suggested by Kapur at. al [14]. Fourier descriptor methods use as shape features the coefficients obtained by Fourier transformation of object boundary [15, 16]. Rui at. ai [17] proposed a modified Fourier descriptor for shape representation which is invariant to geometric transformation and robust to noise. We use the following shape feature computed over the segmented object in our system.

Circularity ratio:

1. circularity1= $Perimeter^2/Area$
   where
   $$Perimeter = \sum_i \sum_j Edgemap(i,j)$$
   $$Area = \sum_i \sum_j Objectmap(i,j)$$
   This is a measure for circularity of the object. For a circle this value is equal to 4π, so the circularity of the object is inversely proportional to the absolute value of the difference between ratio and 4π. We divide this result by 4π to get normalized result. This feature is independent from *rotation,translation* and *scaling.*

2. circularity2=$Area/Circulararea$
   To determine Circular area we first have to determine the centroid of the object. Let $(\overline{x}, \overline{y})$ be the centroid of the object. We calculate this by the following formula:
   $$\overline{x} = \sum_i \sum_j i * Objectmap(i,j)/Area$$
   $$\overline{y} = \sum_i \sum_j j * Objectmap(i,j)/Area$$
   This is another measure for circularity of the object. Circular area is the area of smallest circle which centered at centroid and bounds the object Clearly this is

normalized, if the object is close to circular then its value will be close to 1. This feature is also independent from *rotation*, *translation* and *scaling*.

Median ratio: We use two different types of median ratio as feature.

1. **Horz_Vert ratio:** This is a normalized estimation of height and width ratio of the object. For every vertical position we find the length of chord from extreme left boundary pixel to extreme right boundary pixel. Take the median among these chords. Similarly do the same for vertical median.
   Take the ratio of these two length in such way that this ratio value become normalized. This feature independent from *translation* and *scaling*, but depends on *rotation*.

2. **Principle ratio:** To determine this ratio we have find first the slope of median axes and slope of cross median axes. We get this by the following formula:
   Let m(i,j) be the (i,j)th moment about centroid, then

$$m(2,0) = \sum_i \sum_j (i - \overline{x})^2 * Objectmap(i,j)$$

$$m(0,2) = \sum_i \sum_j (j - \overline{y})^2 * Objectmap(i,j)$$

$$m(1,1) = \sum_i \sum_j (i - \overline{x}) * (j - \overline{y}) * Objectmap(i,j)$$

   if θ be the angle which the median axis makes with the positive direction of x-axis, then the slope of the median axis tanθ is given by

$$tan^2\theta + ((m(2,0) - m(0,2))/m(1,1)) * tan\theta - 1 = 0$$

   second root of this equation is the slope of the cross-median axis. Now the coordinate of centroid and slopes of median and cross-median axes are known. As like the previous scan Objectmap along the slope of median axis and determine the value of median chord, and do the same for cross-median axis. Then take the ratio of these two value so that ratio becomes normalized. This ratio is independent from *rotation*, *translation* and *scaling*.

Aspect ratio: We already know the coordinate of centroid and slopes of median and cross-median axes . Find the length of these two axes and take ratio of them so that ratio value becomes normalized. This feature is independent from *rotation*, *translation* and *scaling*.

$$aspect = \frac{length\ of\ median\ axis}{length\ of\ cross_median\ axis}$$

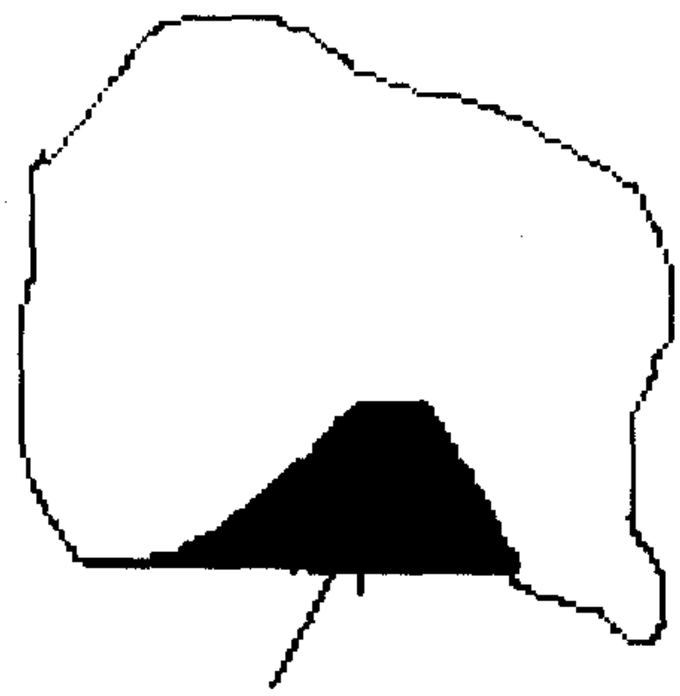


Figure 5: Vacant area of an image

Vacant ratio: Suppose *Varea* is the area of concave region seen from the bottom see Figure 5. Then the Vacant ratio is defined as

$$Vacant\ ratio = \frac{Varea}{Varea + Area}$$

This Is a distinguishing feature particularly for animal in standing pose.

Concavity ratio: Consider the Edgemap of objects and apply Graham Scan [27] algorithm to get convex hull of boundary points of the object. Some results are shown in Figure ??. Then define:

1. **Concavity1:**

$$concavity1 = (convex\ area - Area)/convex\ area$$

This feature is independent from *rotation, translation* and *scaling.* Here *Area* means area of the object of image file and *convex area* means the area of the convex hull.

2. **Concavity2:**

$$concavity2 = convex\ perimeter/Perimeter$$

This feature is independent from *rotation, translation* and *scaling.* Here *Perimeter* means perimeter of the object of image file and *convex perimeter* means the perimeter of the convex hull.

signature related fature

Signature feature is also a shape feature computed from the conture of objects. A signature is a 1-D functional representation of a boundary and may be generated iv various ways. One of the simplest is to plot the distance from the centroid to the boundary as a function of angle.

Regardless of how a signature is generated, however, the basic idea is to reduce the boundary representation to a 1-D function, which presumably is easier to describe than the original 2-D boundary.

Signature is invariant to translation, but they depends on rotation and scaling. Normalization with respect to rotation can be achieved by finding a way to select the same starting point to generate the signature, regardless of the shape's orientation. One way to do so is to select the starting point as the point farthest from the centroid, if this point happens to be unique and independent of rotational aberrations for each shape of interest. Another is to select the point on the principal eigen (major) axis farthest from centroid.

In our project we consider the farthest point on principal eigen (major) as the starting point for signature. We consider 64 boundary points with equal angular spaced. Then applying FFT on these set of data we consider 8 output data (real and imagenary) from 2nd onwards. We use these as signature feature for the images. When images are added to the database these features values are extracted and stored into database.

3.1.3 Texture Feature

Texture is also an important feature in pattern recognition and image retrieval. It refers to the visual pattern of regions of an image. There are several properties of texture, which include *contrast, coarseness, directionality, regularity, periodicity,* and *randomness.* The texture of an image can be represented by Wave-let transform [18] [19]. The ability to retrieve images on the basis of texture similarity may not be very useful. But the ability to match on texture similarity can often be useful in distinguishing between areas of images with similar color (such as sky and sea, or leaves and grass). A variety of techniques has been use for measuring similarity; the best-established rely on comparing values of what are known as *second-order statistics* calculated from query and stored images. Essentially, this calculate the relative brightness of selected pair of pixels from each image. From this it is possible to calculate measure of image texture such as the degree of *contrast, coarseness, directionality* and *regularity* [20], or *periodicity, directionality* and *randomness* [21]. Alternative methods of texture analysis for retrieval include the use of Gabor filters [18] and fractals [22]. Texture query can be formulated in a similar manner to color queries, by selecting examples of desired textures from a palette, or by supplying an example query image. In our project we haven't use any texture feature.

3.2 Multidimensional Indexing

A significant limitation of current CBIR technology is the problem of efficiently retrieving the set of stored images most similar to a given query. One of the many fundamental ways in which CBIR differs from text retrieval is that it is based on fundamentally different model of data. Most text retrieval systems associate each document with a variable number of descriptors representing its content. A given descriptor is either present or absent in a given document. Searching essentially consists of identifying those documents associated with a given descriptor, and is thus governed primarily by the rules of symbolic logic. In such systems, search efficiency can be increased by the use of devices such as *inverted file indexes,* each of which holds a list of document identifier associated with a given descriptor. Boolean searches can readily be implemented by comparing the indexes for each search term, yielding results sets which can be used directly to address the documents themselves.

Most current CBIR systems work on a completely different principle. Stored images are typically characterized by fixed-length real-valued multicomponent feature vectors, each image having a value for every feature in the database. In this case, searching consists of calculating the similarity between feature vectors from query and stored images, a process of numerical computation. The prime aim of CBIR systems is to *sort* the database in order of similarity to the query. Finding index structure which allows efficient searching of an image database is still an unsolved problem. None of the index structures proposed for text retrieval has proved applicable to the problem. Indexing should be in such a way that retrieval of the images can be done efficiently. Multidimensional indexing structure depends on search strategy. An indexing structure can be characterized as follows:

- **Dynamic/Static**: A dynamic indexing structure supports dynamic index updation: insertion and deletion can be intermixed with queries. Static indexing structure not allow any insertion, deletion after indices of the given database created.

- **Memory/Disk resident**: In first case indexing structure creates indices in computer main memory; while in second case indexing structure stores indices on hard disk. In second case a large scale of data set can be accommodate and this structure is also non-volatile.

- **High/Low dimensional**: ISAM or B-tree indexing method can index only 1-D data points. There are so many recent indexing techniques such as R-tree can index higher dimensional points.

From the aspect of the requirements, for CBIR indexing structure must have following characteristics: *dynamic,disk-resident* and *higher-dimensional.* Indexing structure should support non-Euclidean similarity measure, because Euclidean similarity measure cannot simulates human perception. Other non-Euclidean similarity, such as histogram intersection, correlation, etc. may be employed. The most promising approach so far has been multidimensional indexing, using structure such as R*-

tree [23], the TV-tree [24],SS^+-tree [25],KDB-tree [26], but the overhead of using these complex index structures are considerable.

We give more emphasis on finding out features of images. For indexing we only stored the features vector of each image as a record in a sequential file. When we add or delete an image our database scheme behaves like sequential file, which is not efficient. We will be take care of this problem in future work.

3.3 Retrieval Engine

Most of the current CBIR systems retrieve images by measuring some kind of similarity. Once the visual features are computed, the images represented by points in multidimensional feature space, and a distance (e.g. Euclidean distance, City block distance and some other distance metric) measure between two points used a similarity metric. There are two types of search schemes:

1. k-nearest search

2. range search

The first scheme, retrieve k images corresponding to k nearest points of the reference (query image). It does not care how similar the retrieved images are. In second case retrieves all the images lying with in a given range.

We use Euclidean distance as similarity metric. It is known that when # of components of a vector is more then for similarity matching Euclidean distance measure gives poor results. We use here k-nearest searching scheme for getting output. Here we consider top 10 output.

4 Implementation of User interface

between two points used a similarity metric. In our previous discussion we already mentioned that interface is one issue of CBIR system. Before the development of the system, the language in which the system has to be develop was an issue. Here language VC++(6) is used . It is desirable that the system must have GUI facilities and for that we developed a form based interface. Several buttons are embedded on the form , some of which are linked to dialogbox and combobox. For GUI interface we use two form MAIN form and QUERY form. These two from are shown in Figure 6 and Figure 7. In MAIN form there are following buttons:

- **Display:** Whenever user clicked this button the image which is selected from combobox will be displayed on appropriate view port.

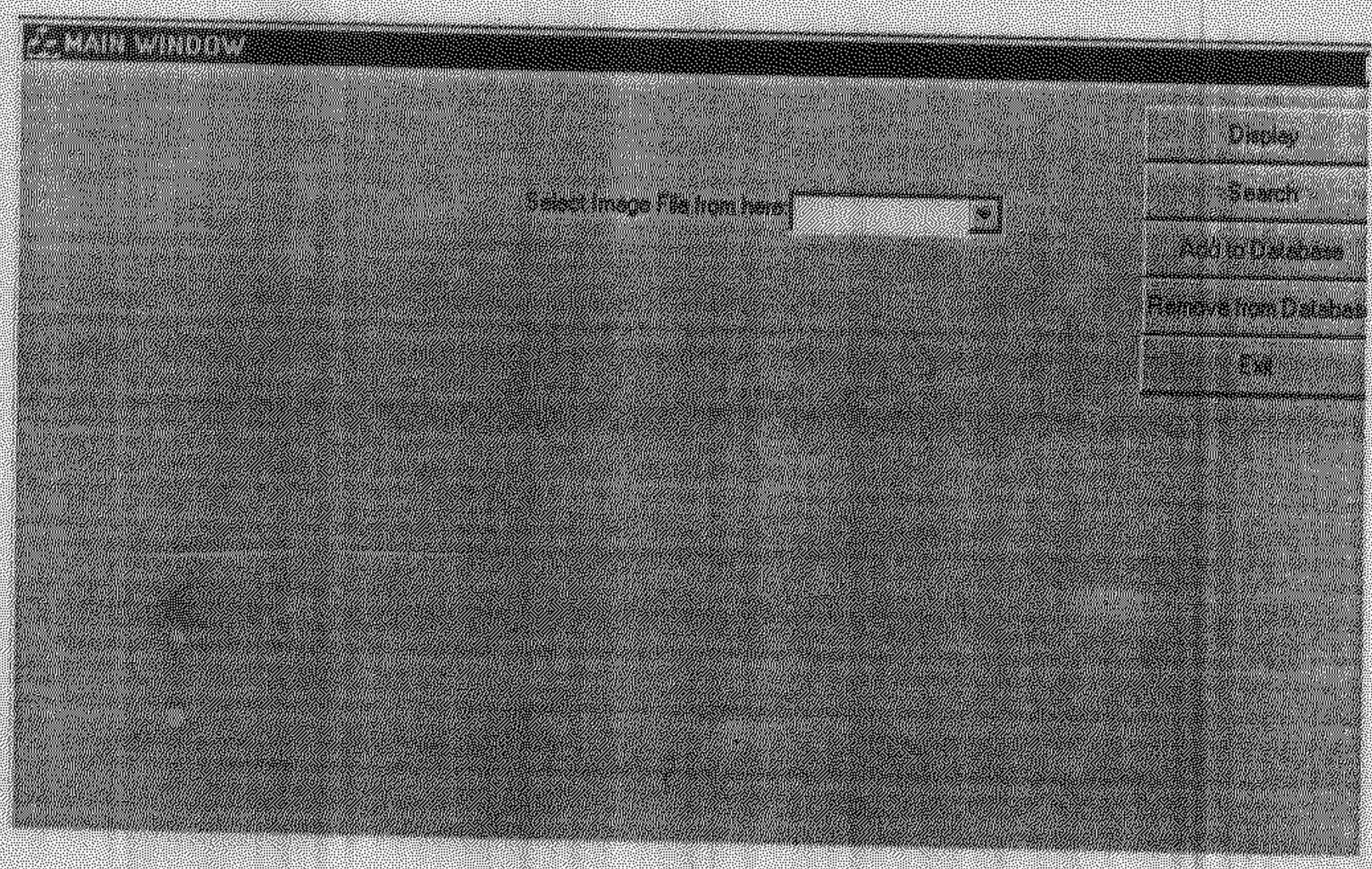


Figure 6: Main form

- **Add to database:** Clicked on this button opens a browse capability similar to the one used when a file is to opened in Windows environment. Only bitmap files will be shown inthis file dialog box.

- **Remove:** The selected file from combobox will be removed from database if this button is clicked.

- **Search:** The button have linked to another dialog box (QUERY). When this buttton is clicked QUERY form will be on the screen and the query image will be display on the left corner of the dialogbox.

In QUERY form there are following buttons:

- **Group Radio button:** There are a pair of radio button which are used for selects the query option,i.e. either *query by shape feature* or *query by color feature*. If we selects one option then the desired output will be displayed on screen.

- Combo Box: In this form there is a Combo Box where the results of query (top 10) stored according to distance.

- Display: Select one image from Combo Box and click Display button, corresponding iage will be displayed on screen.

- **Return:** This button is used to return back to the MAIN form.

In our project we have tried to develop a CBIR system having some characteristics which are:

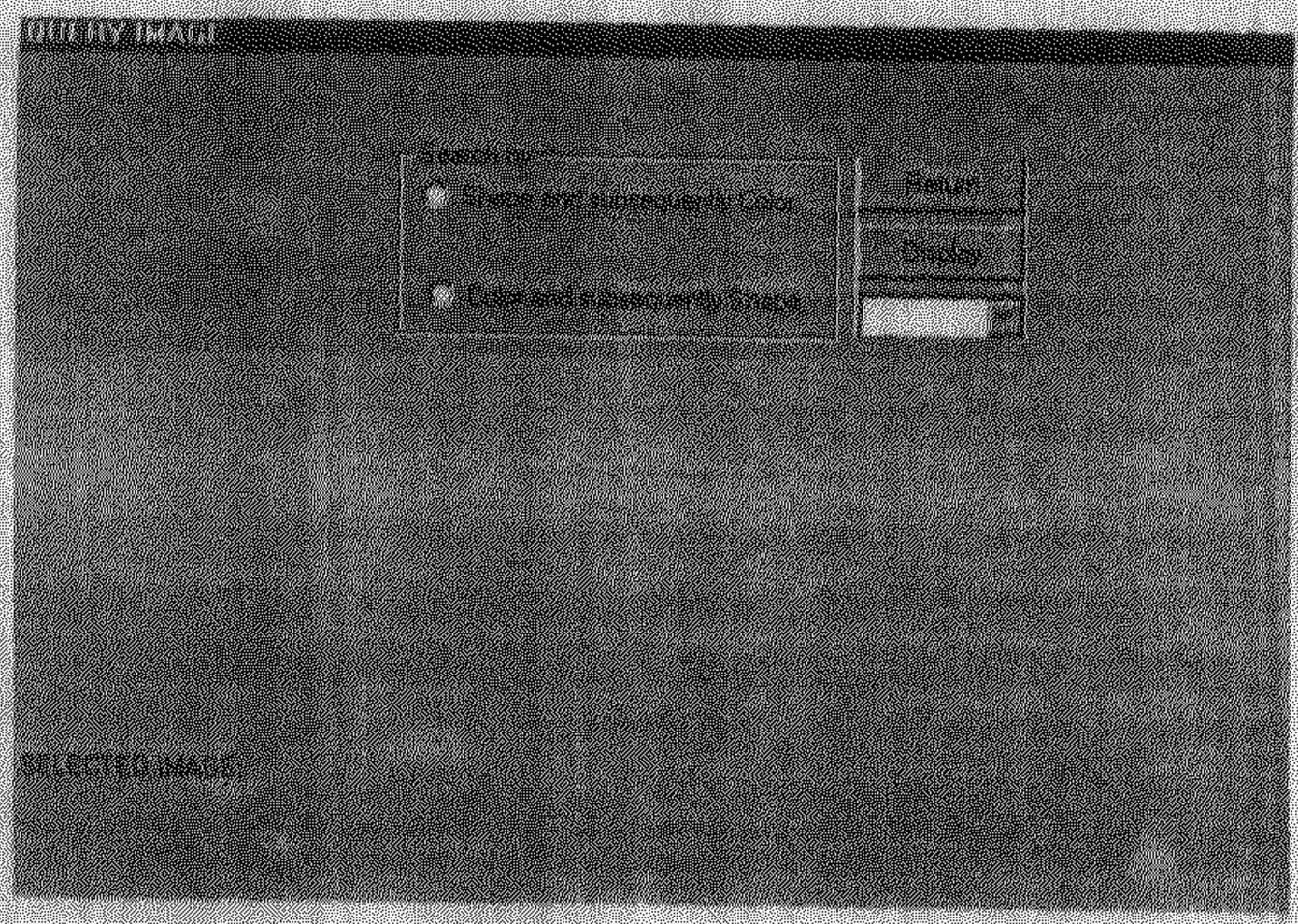


Figure 7: Query form

- Provides a graphical user interface to the user through which user can add, remove, display and search images.
- Here only 24-bit bitmap image files are used.
- It is a Content Based Image Retrieval System, where the content of images are color, shape of the objects.
- There is search mechanism through which user can pose a query from within database, the result of the query (top 10) will be displayed on screen.
- There is also facilities of the selection search option, i.e. either by shape or color.
- To add a new image to the database, browsing facility also available.
- This is a two in one system. The system has capabilities of both a color matching and shape matching. Both the color and shape features of the object in the image file are extracted when an image file is added to the database. These features are stored in from of a record in file. Similarly when an image file is removed from the collection, the record corresponding to its feature vector is removed from the file.

5 Experimental Results

We have implement our system on Pentium based hardware and Windows as operating system with VC++(6) used as language. The number of images we used in our study

Birds	Air-craft	Flower	Mammals	Others
156	20	26	95	15

Table 1: Number of distinct images used

is 313. They are collected from various sources such as downloaded from web, scanned from hardcopies, image acquition using digital camera, etc.. The images consists of flowers,birds, mammals, air-craft, etc.. Some example images are shown in Figure 8. In our database we used 313 images and a list of image content and their number is given in Table 1.

On the hard disk all these images, along with their feature vectors are stored. We pick up the query image from the same database and find its best 10 matches. To do this,

1. First calculate the distance (Euclidean) of each images from the query image.
2. Sorts these images w.r.t. distance.
3. Selects top 10 images (which are more similar).
4. Display these images on screen.

Some of the results of such retrieval are shown in Figure 9 and Figure 10.

5.1 Performance Analysis

To evaluate the performance of the system (i.e., combination of extracted features and distance metric) we used to shown how many retrieval images come from the same group to which the query image itself belongs. Since, we don't have the groundtruth of grouping, we simple use K-means algorithm to partitioned images into groups. Here we portioned into 8 groups. After this grouping we use all images as test data as well as trainee data ,then calculate performance and we see that 82.59% success on an average (i.e., on an average after giving an image as query image 82.59% of output images come from same group of query image).

Next we randomly selects $\frac{1}{3}$ images, and used them as test data and remaining images are trainee data. we repeat this process 10 times. We calculate the performance the system, we get 76.15% success. From this we can that this grouping is more or less stable.

6 Conclusion

In our project we try to implement a CBIR system. In our system we use both shape feature and color feature. We haven't consider texture feature. We should consider this

as feature; we will use this in future.Though we also use signature value also as shape feature, but as segmentation is not so-good this feature does not gives any facilities. So for this needs a good segmentation algorithm. We will take care about this in our future work. We store images and its feature in hard disk. We use sequential file to store feature vectors. This will be time consuming when image size is very large. For efficent accessing we have to use multidimensional indexing scheme. Here we use Euclidean distance as similarity metric, but it is known that when number of component is large then this metric gives poor results. So to get better results it is needed to chnage the similairty measurement function.

Our future plan related to this system:

- Consider texture feature.
- Multidimensional Inedixng Scheme.
- Similarity Metric.
- Better Segmentation Algorithm.



Figure 8: Some used image (contd.)

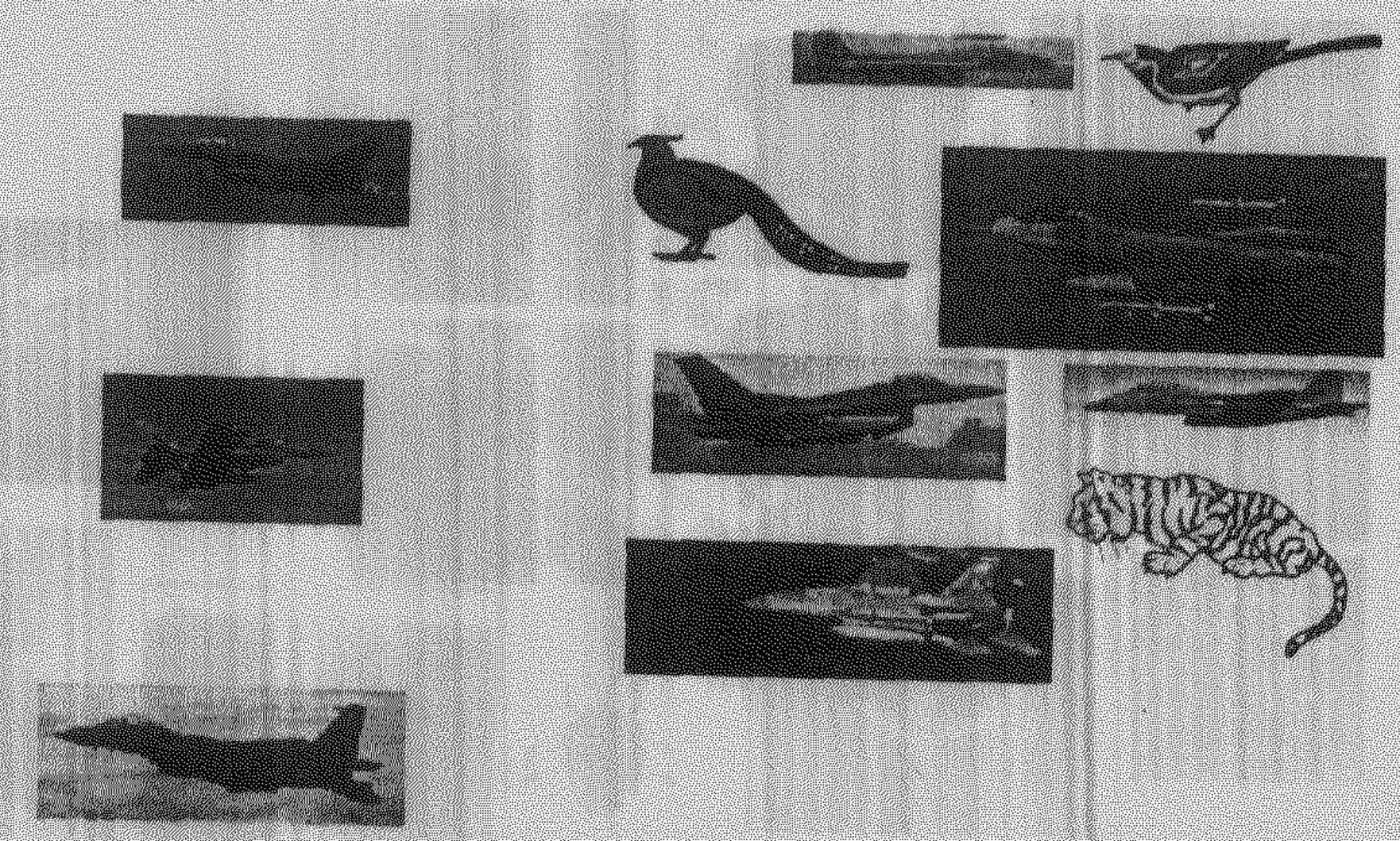


Figure 8: Some used image (contd.)

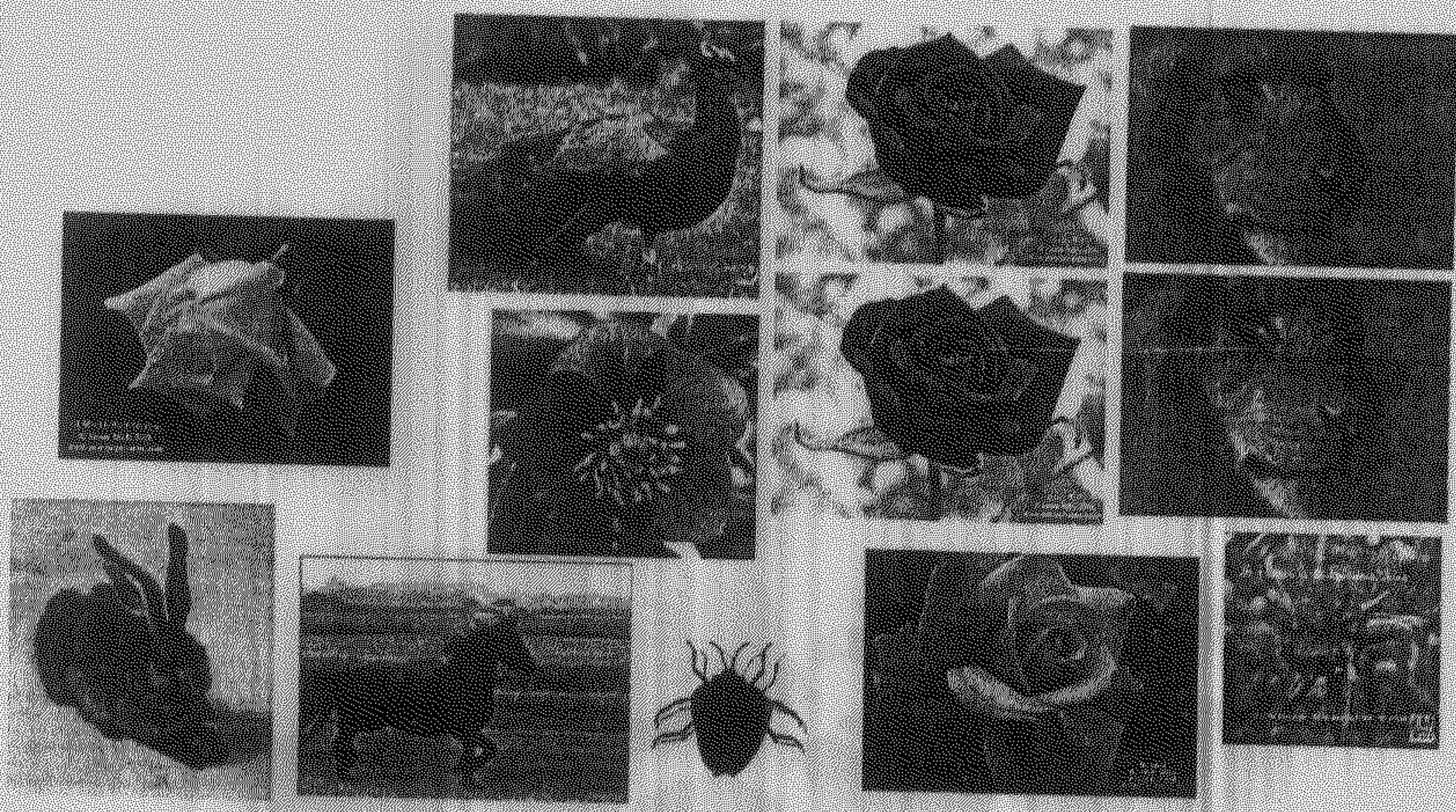


Figure 8: Some used image (contd.)



Figure 8: Some used image (contd.)



Figure 8: Some used image (contd.)



Figure 8: Some used image (contd.)

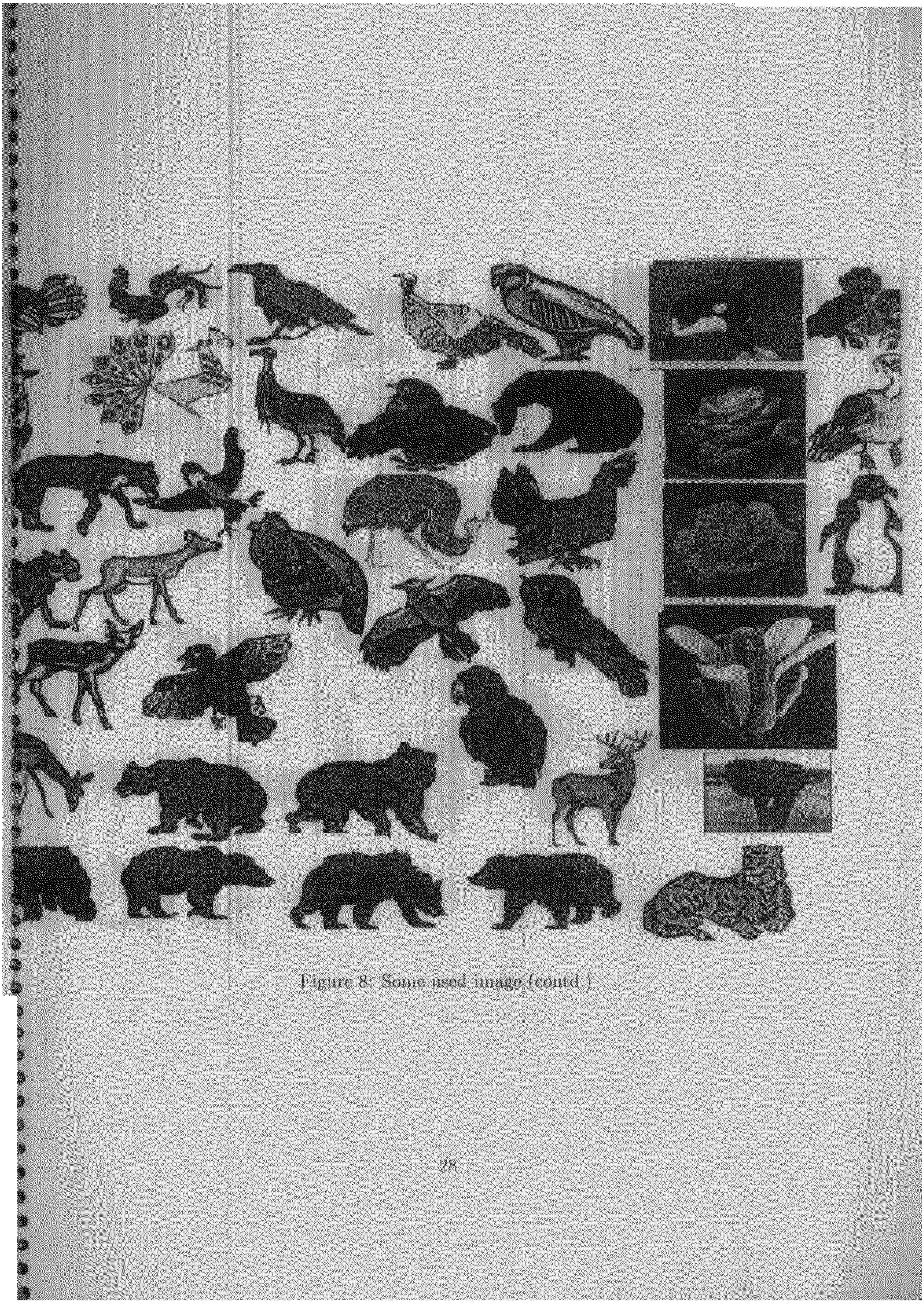


Figure 8: Some used image (contd.)

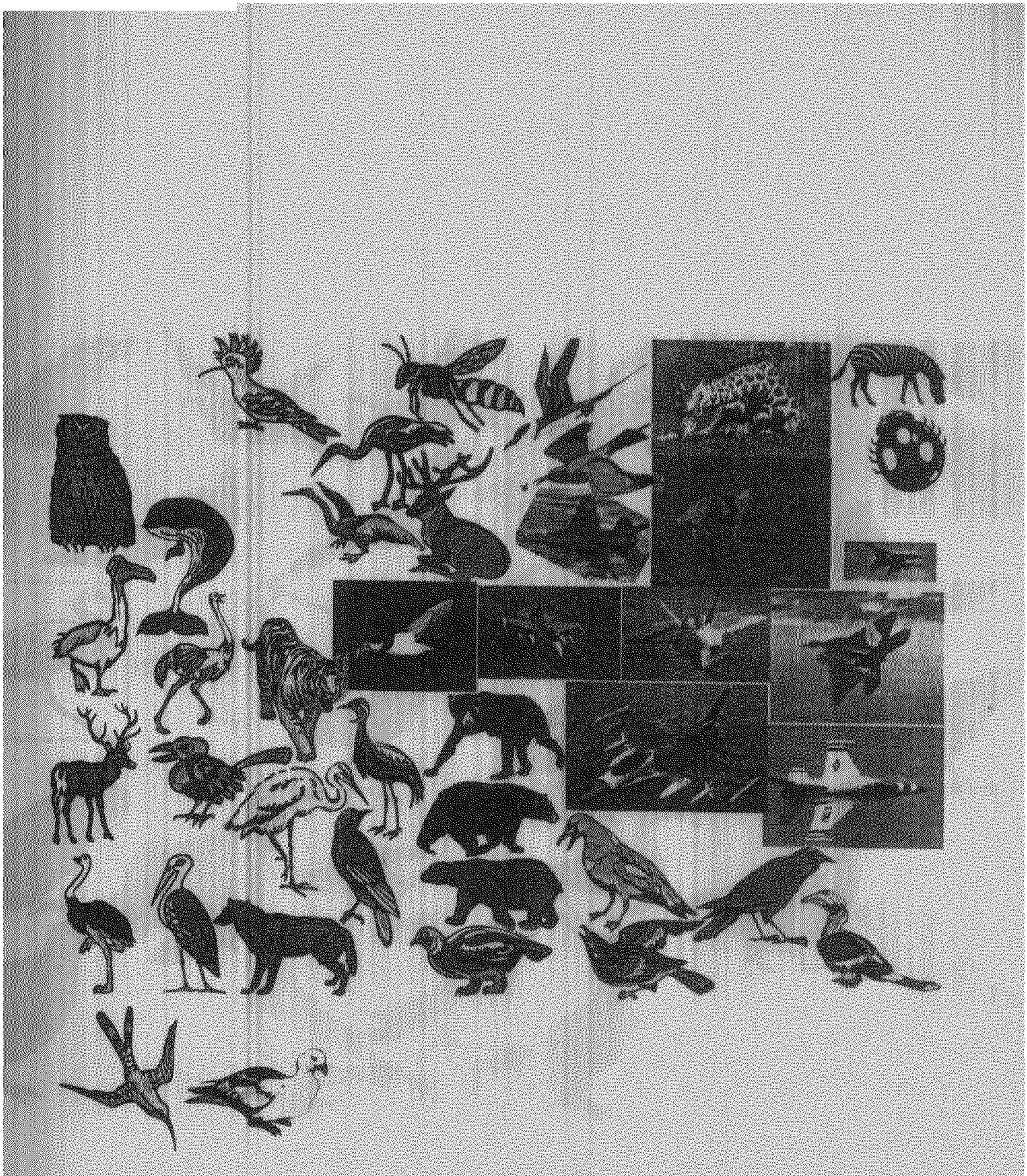


Figure 8: Some used image (contd.)

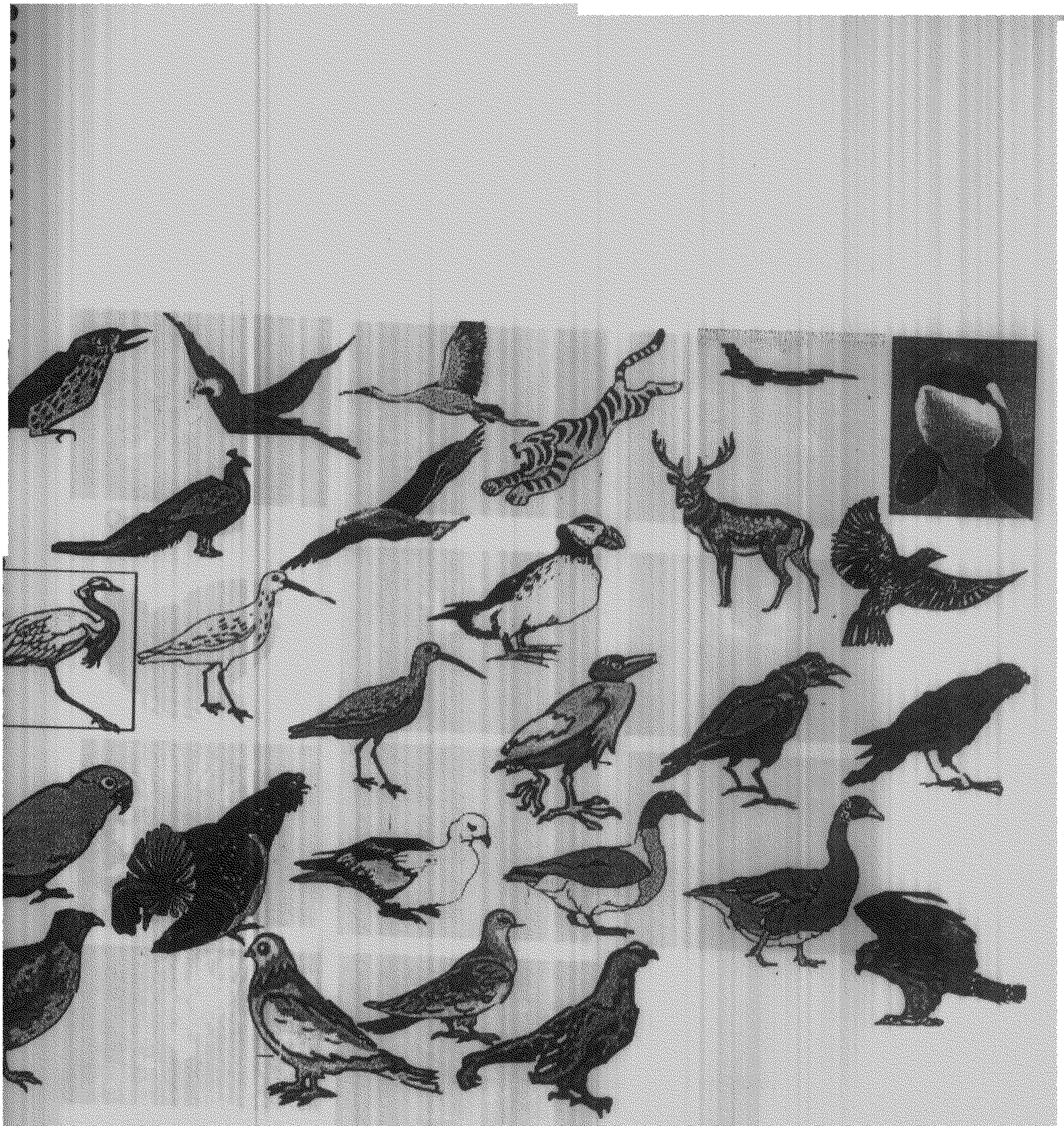


Figure 8: Some used image

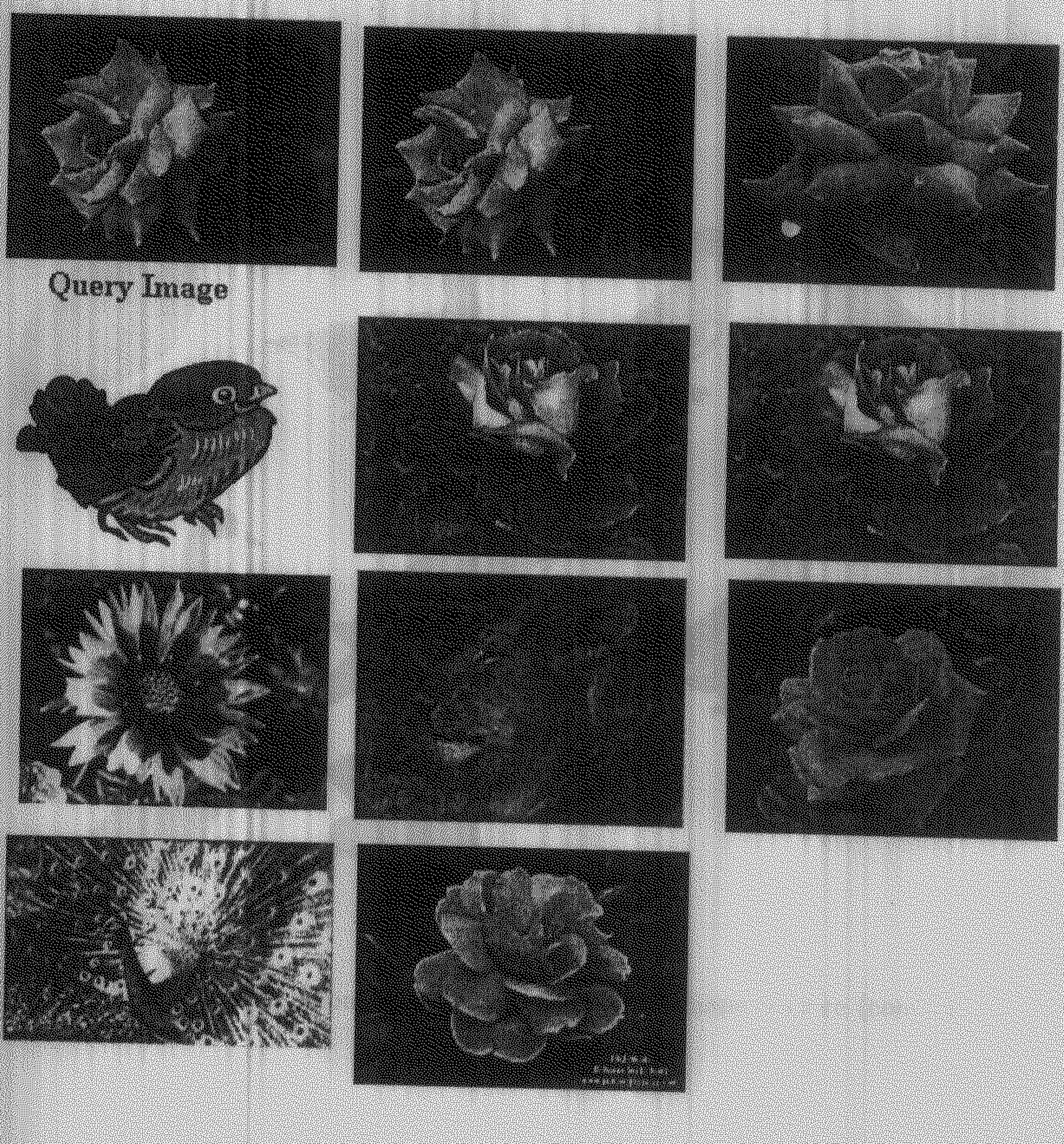


Figure 9: Output of the system based on shape feature (contd.)

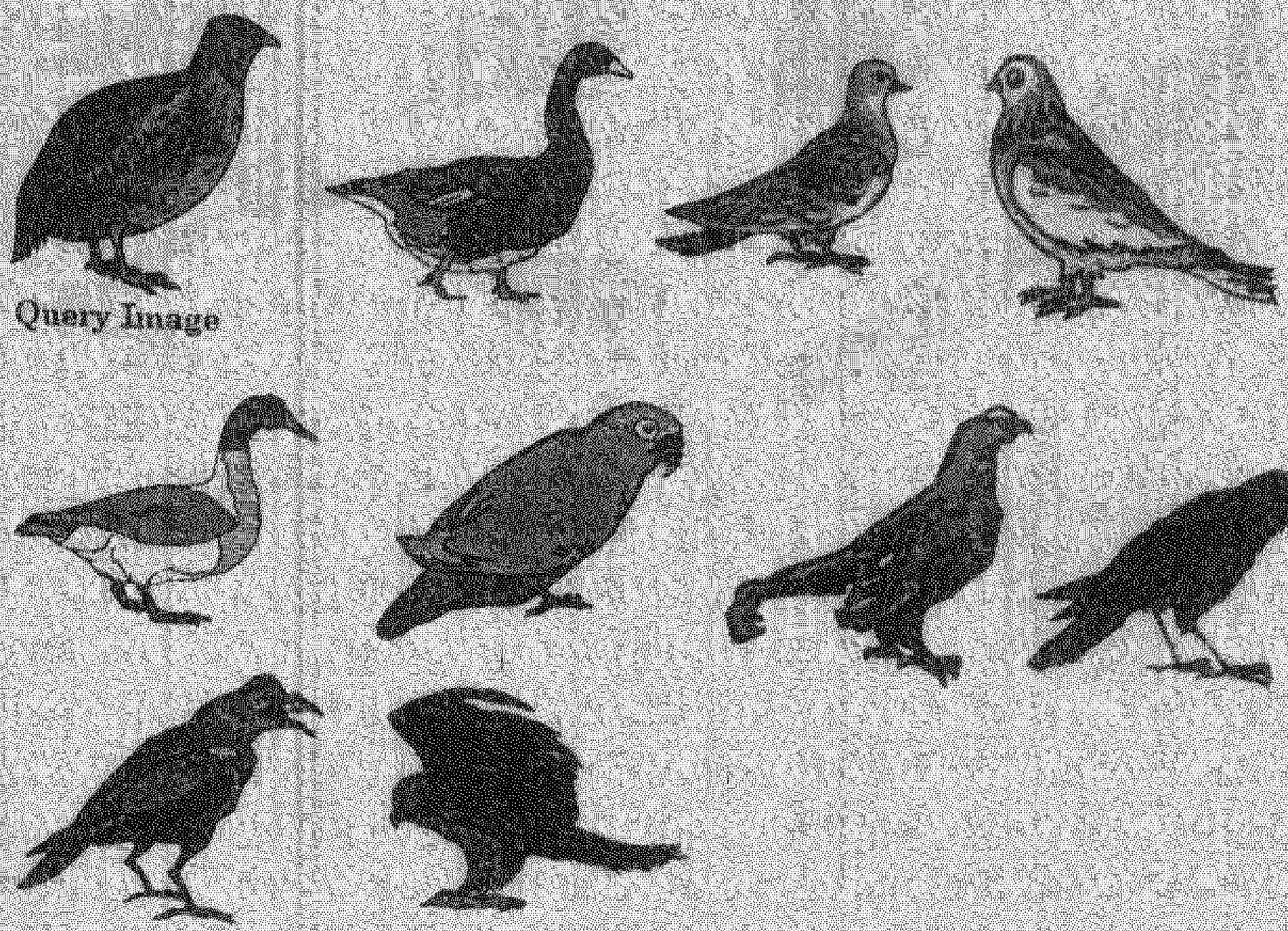


Figure 9: Output of the system based on shape feature (cont.d.)

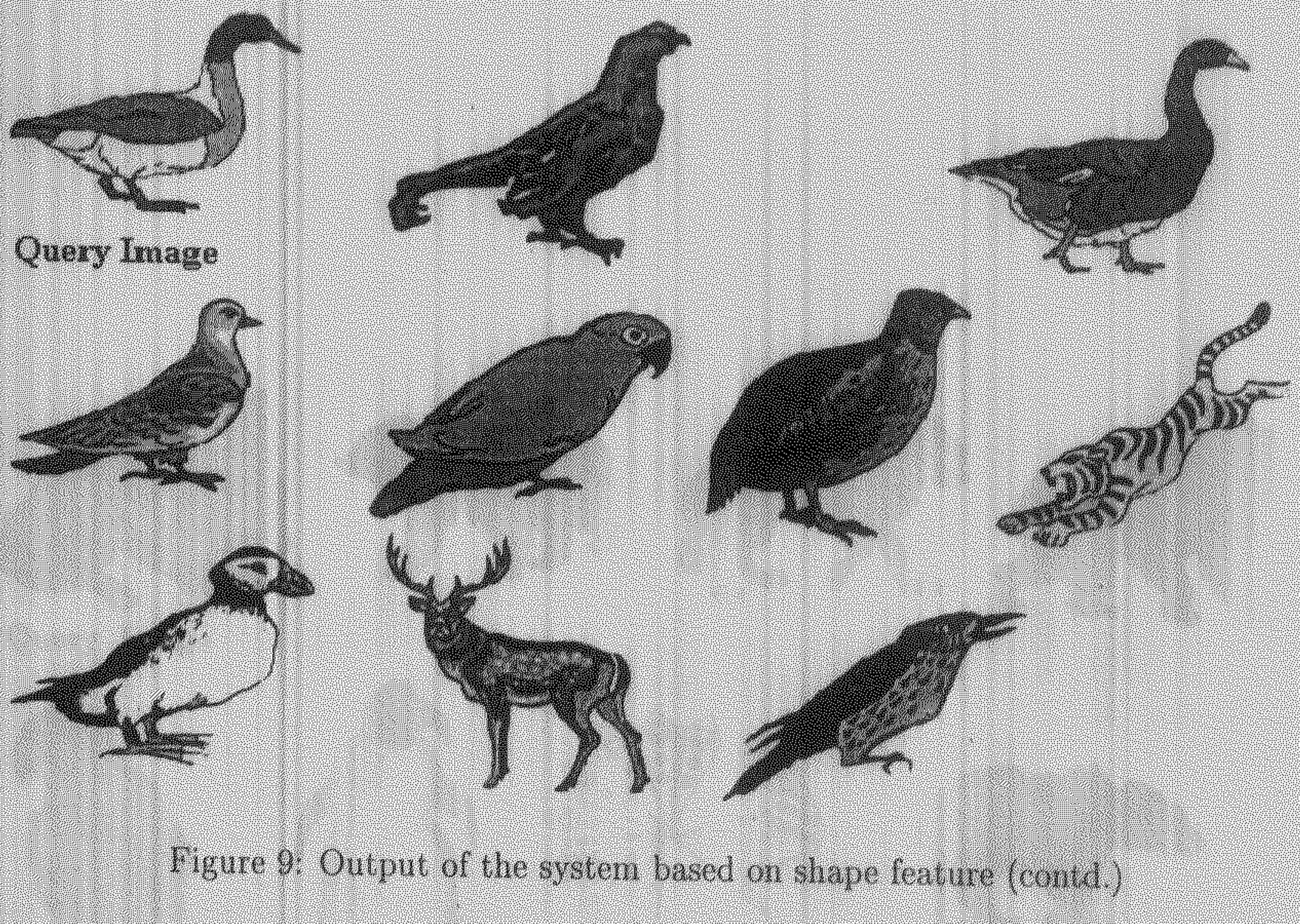


Figure 9: Output of the system based on shape feature (contd.)

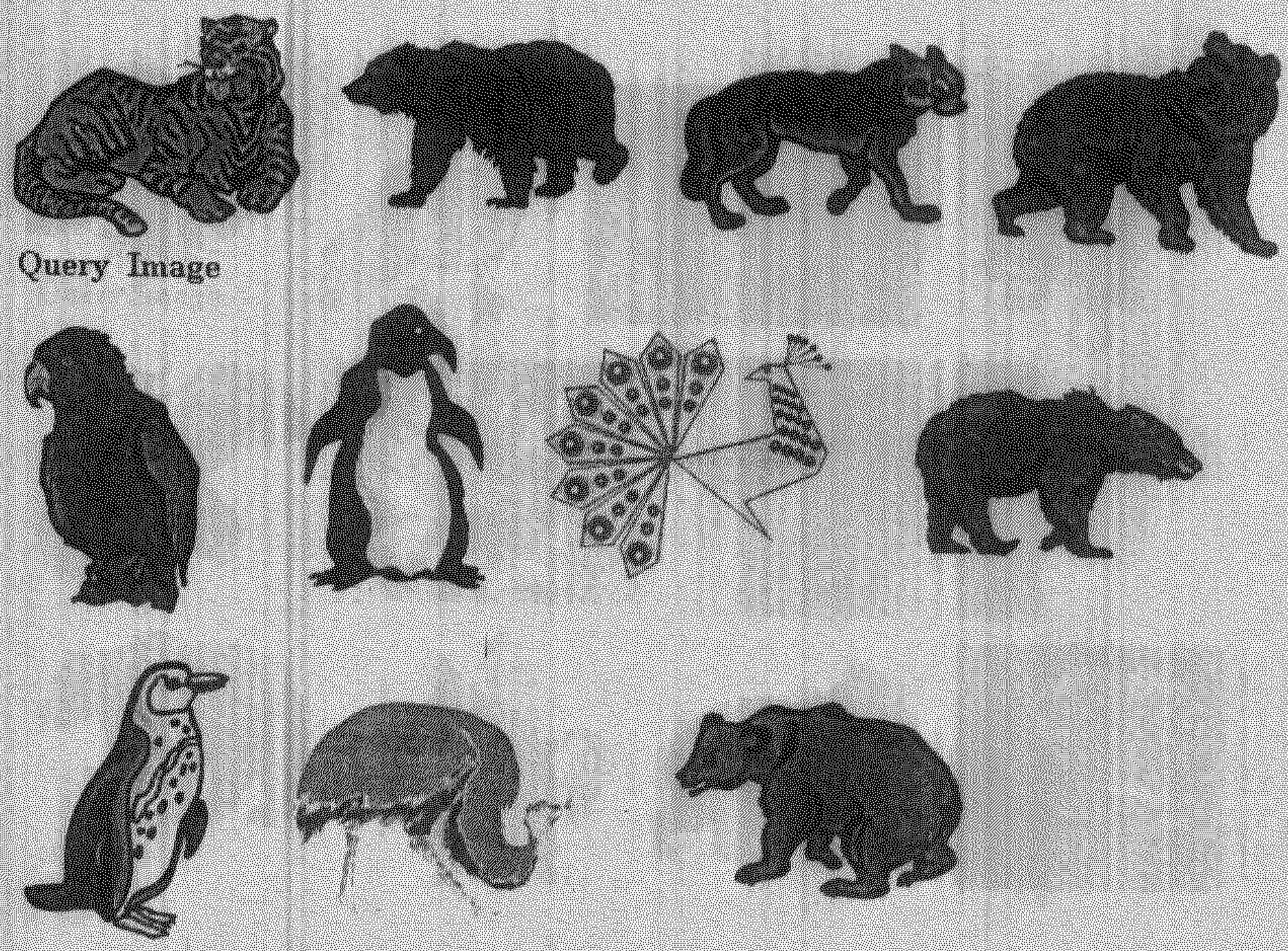


Figure 9: Output of the system based on shape feature (contd.)

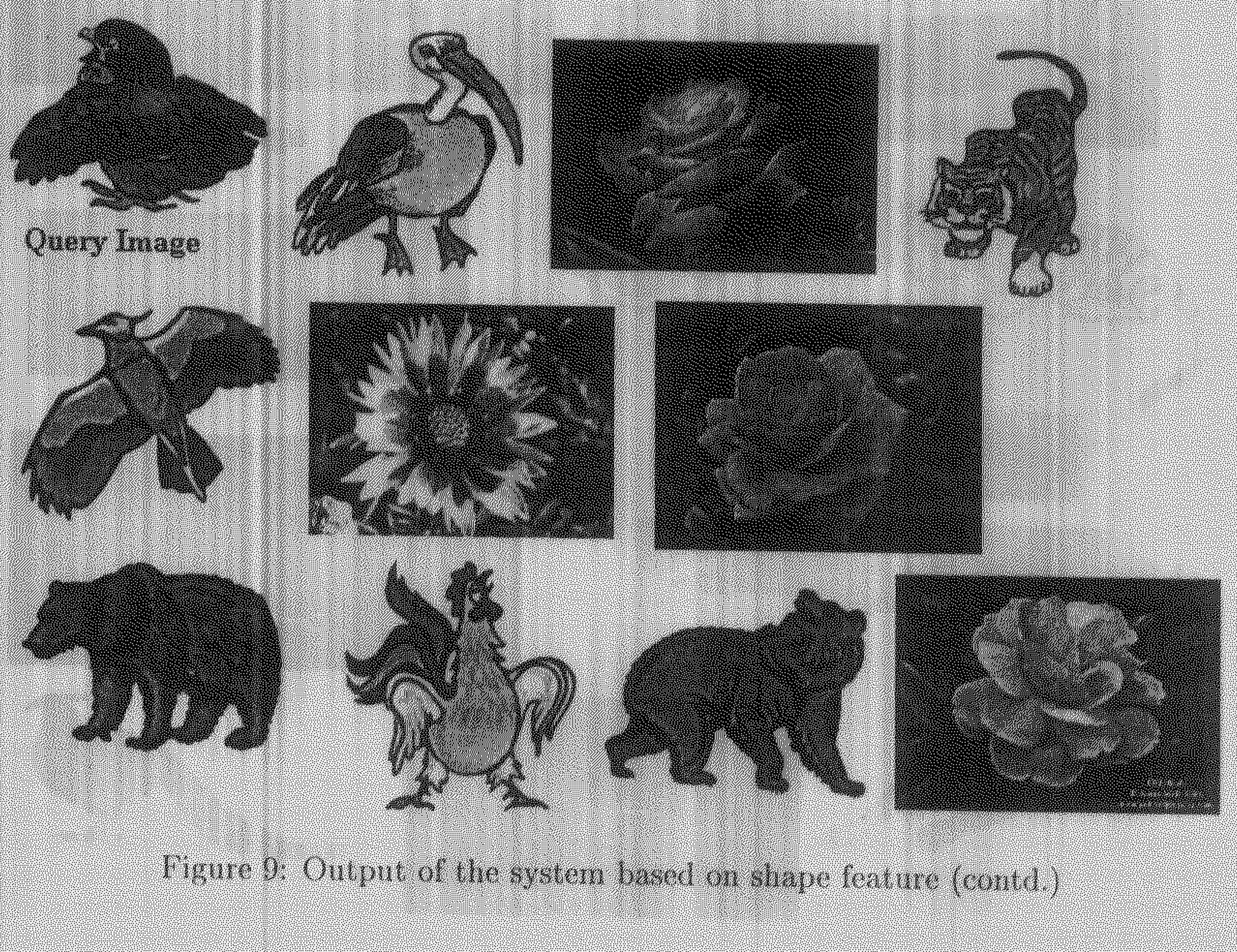


Figure 9: Output of the system based on shape feature (contd.)

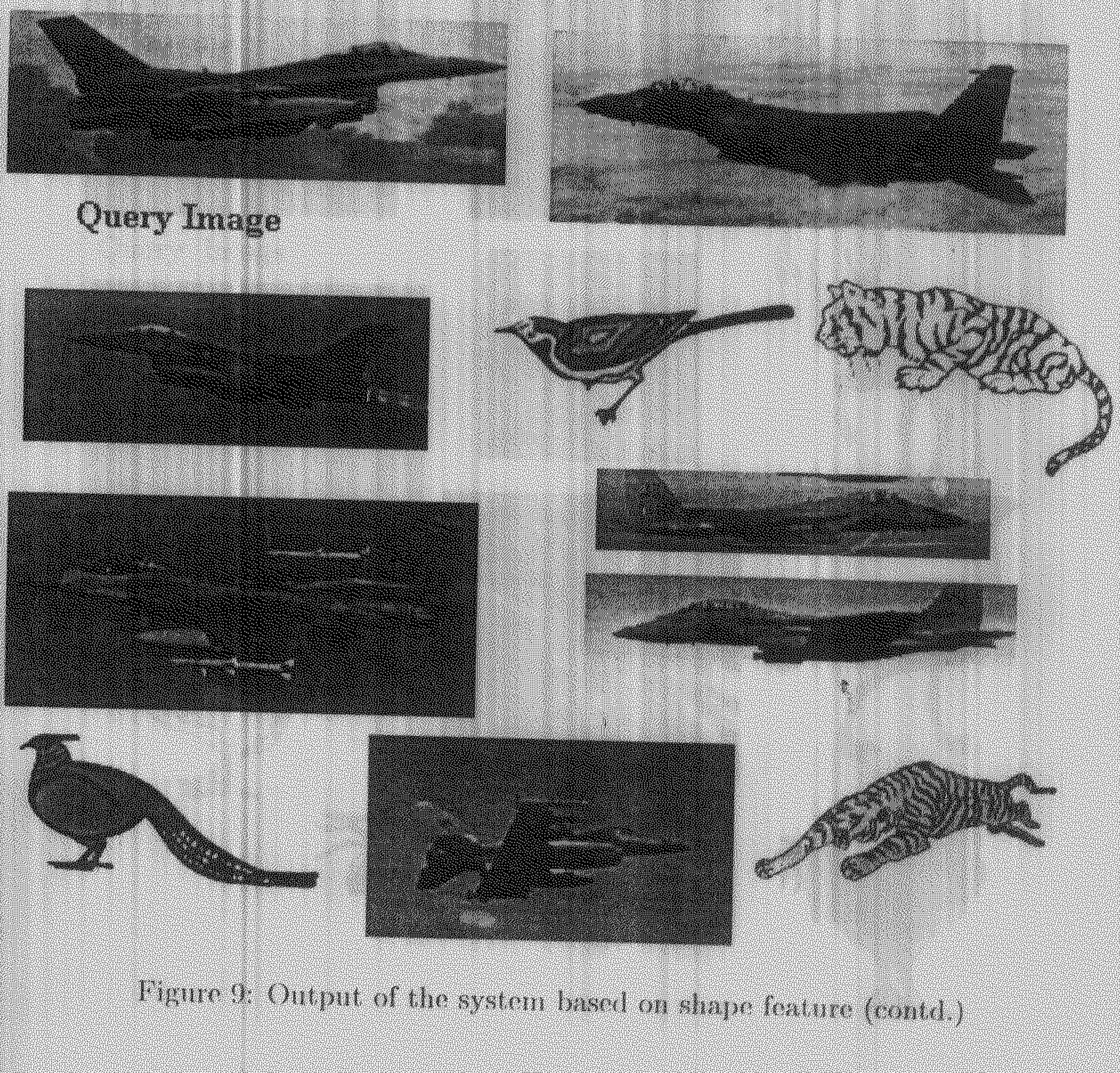
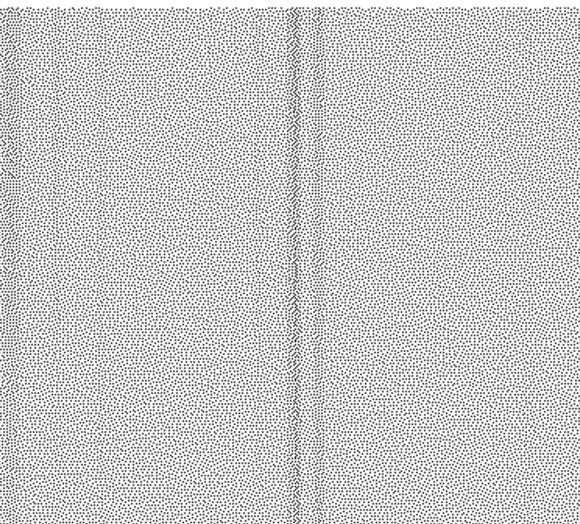
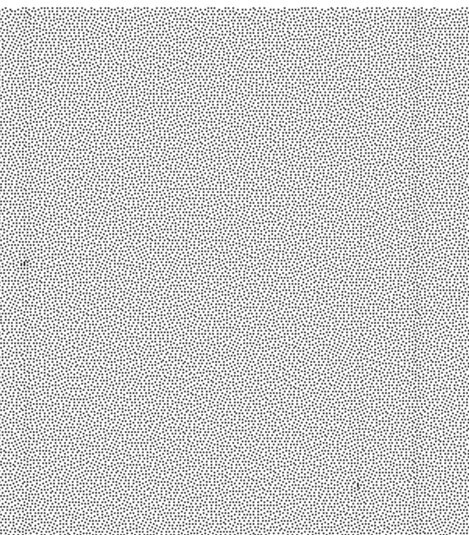
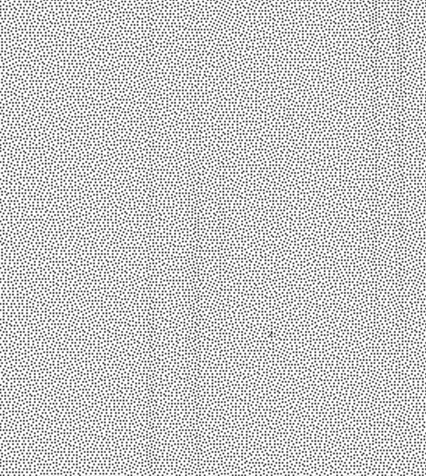


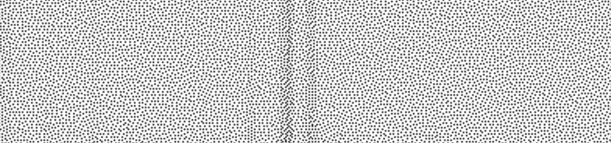
Figure 9: Output of the system based on shape feature (contd.)

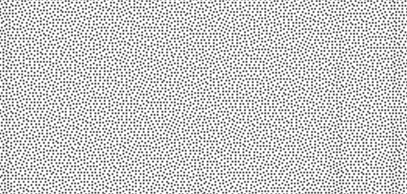


Query Image









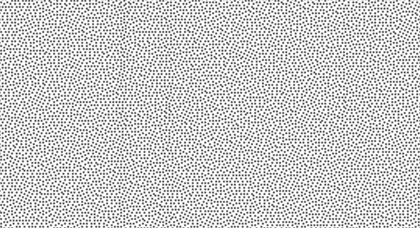










Figure 9: Output of the system based on shape feature (contd.)

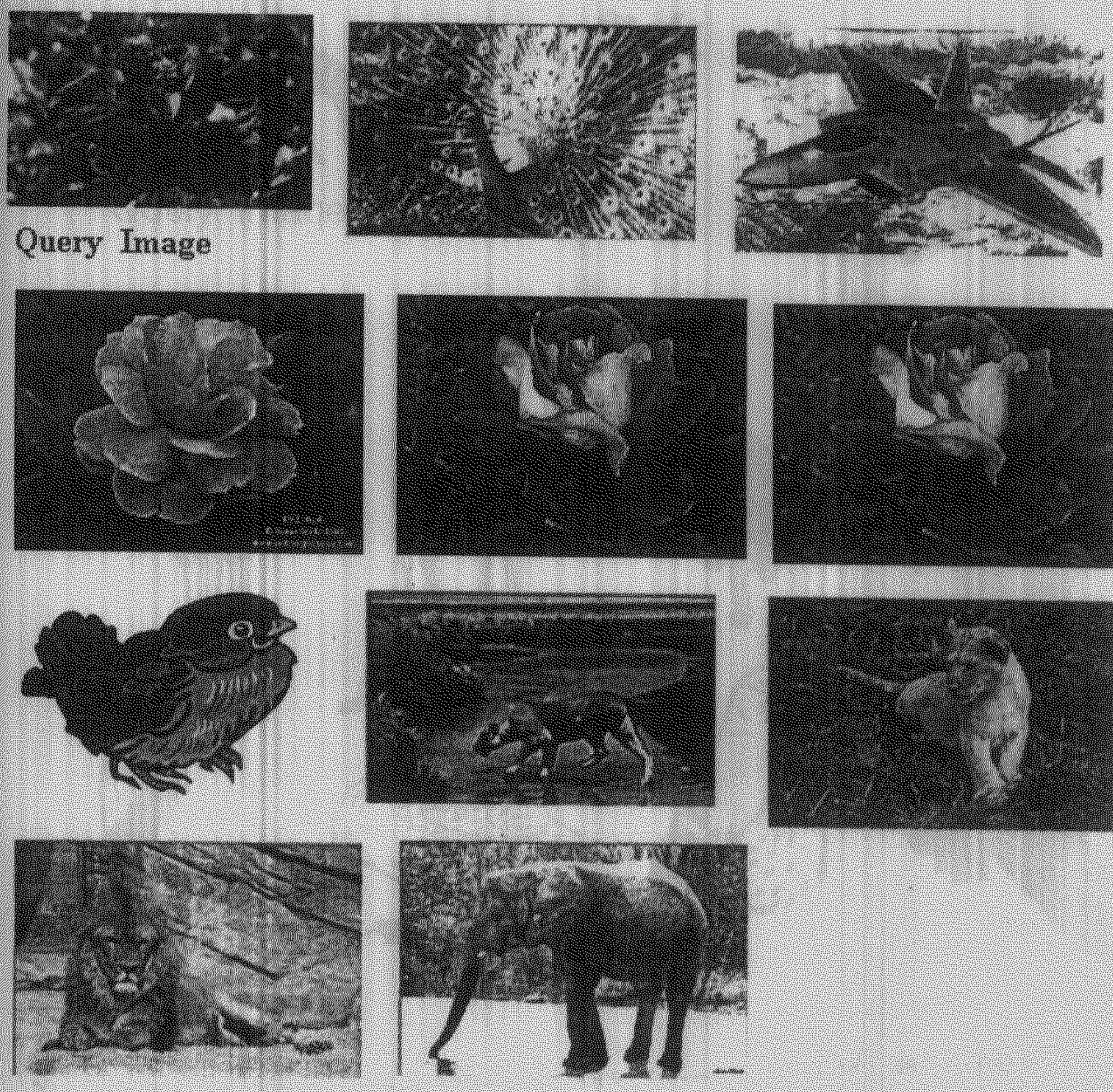


Figure 9: Output of the system based on shape feature



Figure 10: Output of the system based on color feature (Contd.)



Figure 4a: Output of the system based on color feature (Cont.)

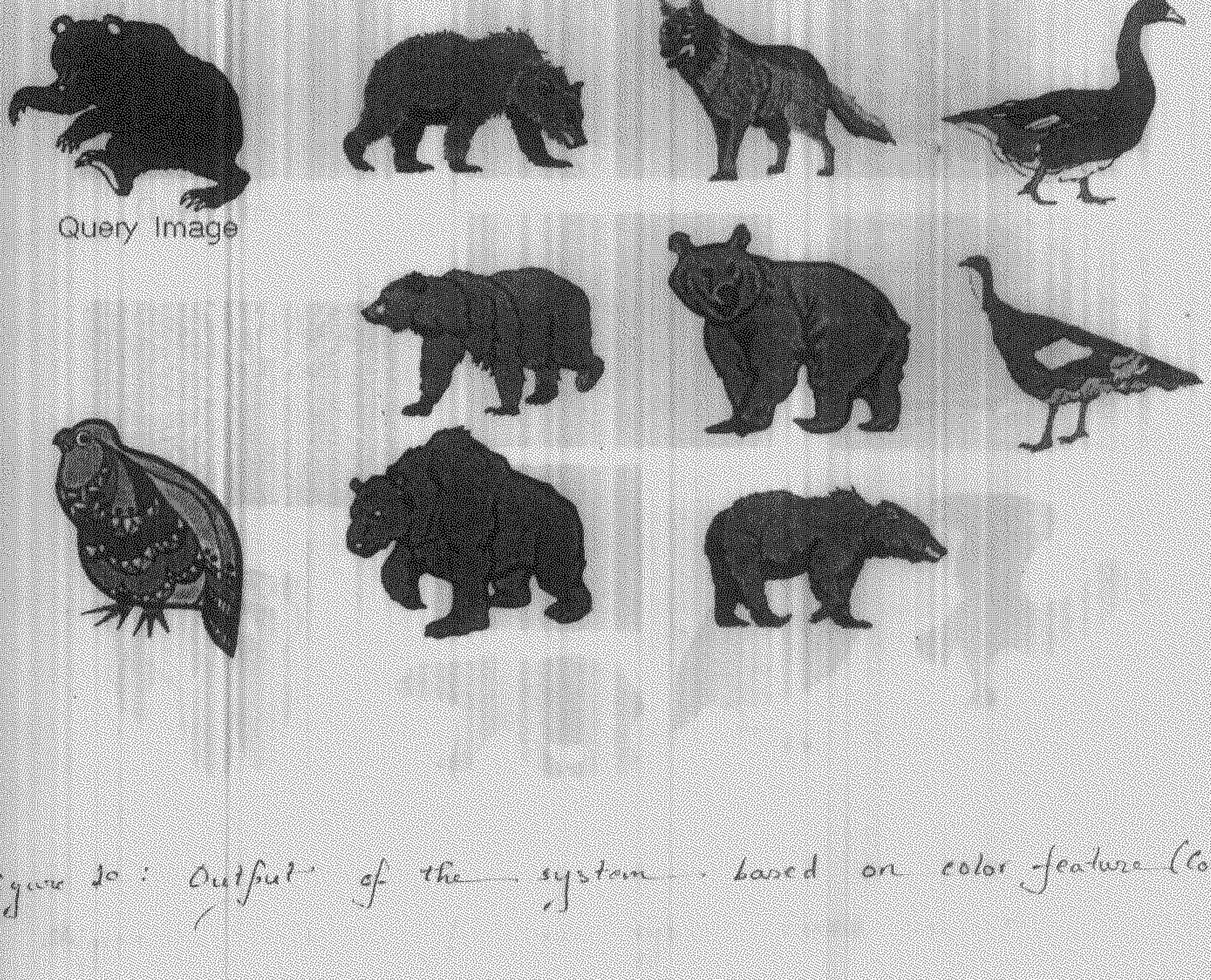


Figure 10: Output of the system based on color feature (Contd.)

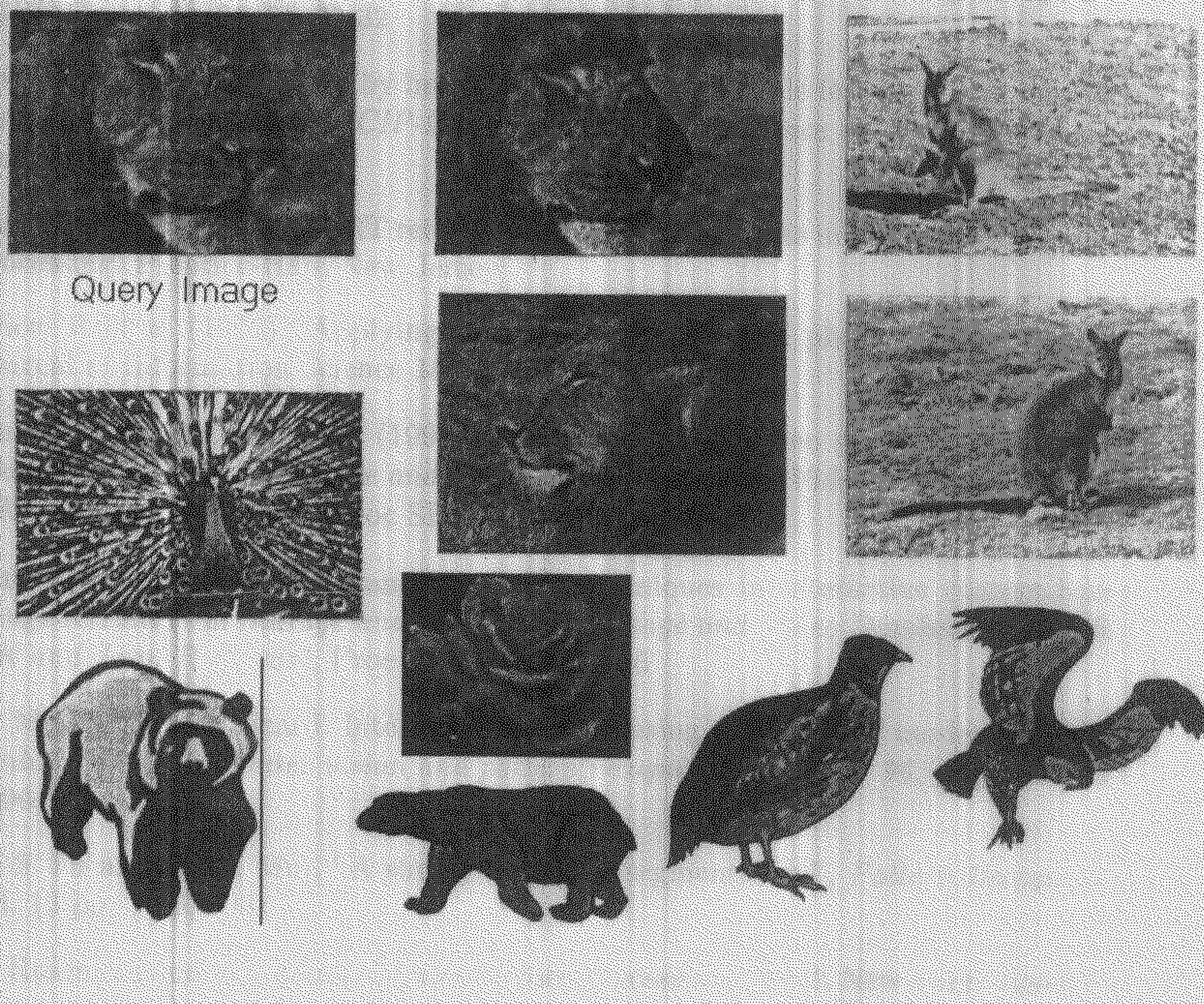


Figure 10: Output of the system based on color feature.

References

[1] R. Jain, "World-wide maze," *IEEE Multimedia,* vol. 2, pp. 3-23, 1995.

[2] Rabbitti, F and Stanchev, P (1989) "GRIM_DBSM: a graphical image database management system" in Visual Database Systems (Kunii, T, ed), Elsevier, Amsterdam, 415-430.

[3] Flincker, M et al (1995) "Query by image and video content: the QBIC system" *IEEE Computer* **28**(9), 23-32

[4] Lee, D et al (1994) "Query by image content using multiple features: user interface issues" in *Proceedings of ICIP-94, International Conference on Image Processing, Austing , Texas*, 76-80

[5] Salton, G (1971) The SMART retrieval system - experiments in automatic document processing. Prentice-Hall, New Jersey

[6] Price, R et al (1992) "Applying relevance feedback to a photo archival system" Journal of Information Science 18(3), 203-215

[7] Smith J R and Chang S F (19997b) "An image and video search engine for the World-Wide Web" in *Storage and Retrieval for Image and Video Database V (Sethi, I K and Jain, R C, eds)*, Proc SPIE 3022, 84-95.

[8] Rui, Y et al (1997) "Relevance feedback techniques in interactive content-based image retrieval" in *Storage and Retrieval for Image and Video Databases VI (Sethi, I K and Jain, R C, eds)*, Proc SPIE 3312, 25-36

[9] T. Minka, and R. Picard. Interative learning using a 'society of models'. **In** *Proceedings of the computer vision and pattern recognition (CVPR) 96,* **pages 447-452,** 1996.

[10] K. S. Fu amd J. K. Mui, "A survey on image segmentation", ***Pattern Recognition***, vol. 13 pp. 3-16, 1981.

[11] A. Rosenfeld and A. C. Kak, *Digital Picture Processing, Vol. 2.* **New** York: Academic Press, 1982.

[12] R. C. Gonzalez and R. E. Wooods, ***Digital Image Processing***. Reading, Mass.: Addison-Wesley, 1992.

[13] B. Chanda and D. DuttaMajumder, ***Digital Image Processing*** *and Analysis*. New Delhi: Prentice-Hall of India, 2000.

[14] D. Kapur, Y. N. Lakshman, and T. Saxena, "Computing invariants by elimination methods", in *Proc. IEEE ICIP*, 1995.

[15] C. T. Zahn and R. Z. Roskies, "Fourier descriptors for plane closed curves", *IEEE Trans. on Computers,* vol. 21 , pp. 269-281,1972.

[16] E. Persoon and K.S. Fu, "Shape Discrimination using Fourier Descriptor", *IEEE Trans. on SMC,* vol. 7 , pp. 170-179, 1977.

[17] Y. Rui, A. C. She, and T. S. Huang, "Modified Fourier descriptors for shape representation - a parctical approach," in *First Int. Workshop on Image Databases and Multimedia Search,* 1996.

[18] B. Manjunath, and W. Ma. Texture feature for browsing and retrieval of image data. In *IEEE Transaction on Pattern Analysisa and Machine Intelligence,* 18(8):37-42, August 1996.

[19] S. Hatipoglu, S. K. Mitra, and K. Kingsbury, Image texture description using complex wavelet transform. In *Proceeding of IEEE International Conference on Image Processing,* vol. 2 pp. 530-533, Vancouver, B. C., Canada, September 2000.

[20] Tamura, H et al, (1978) "Textural features corresponding to visual perception," in *IEEE Transaction on System, Man and Cybernetics* 8(6), 460-472.

[21] Liu, F and Picard, R W (1996) "Periodicity, directionality and randomness: Wold features for image modelling and retrieval" *IEEE Transaction on Pattern Analysisa and Machine Intelligence,* 18(7),722-733.

[22] Kaplan, L M et al (1998) "Fast texture database retrieval using extended fractal features" in *Storage and Retrieval for Image and Video Databases VI(Sethi, I K and Jain, R C, eds),* Proc SPIE 3312, 162-173

[23] Beckmann, N (1990) "The R*-tree: an efficient and robust access method for points and rectangles" *ACM SIGMOD Record* 19(2), 322-331

[24] Lin, K I et al (1994) "The TV-tree: an index structure for high dimensional data" *Journal of Very Large Databases* 3(4), 517-549

[25] Kurniawati, R et al (1997) "The SS^+ tree: an improved index structure for similarity searches in high-dimensional feature space" in *Storage and Retrieval for Image and Video Databases V (Sethi, I K and Jain, R C, eds)*, Proc SPIE 3022, 110-120

[26] J. T. Robinson, "The k-d-b tree: A search structure for large multidimensional dynamic indexes," in *Proc. ACM SIGMOD,* (Boston), June, 1984.

[27] T. Cormen, C. Leiserson and R. Rivest *Introduction to Algorithms* New Delhi: Prentice-Hall of India, 2000, page 899-905.