# Automated Generation of Keywords from Images for Geometric Information Search


Dan Song

LMIB-School of Mathematics and Systems Science,
Beihang University, Beijing 100191, China

**Joint work with Xiaoyu Chen and Dongming Wang**

songdan151100@163.com


June 29, 2016

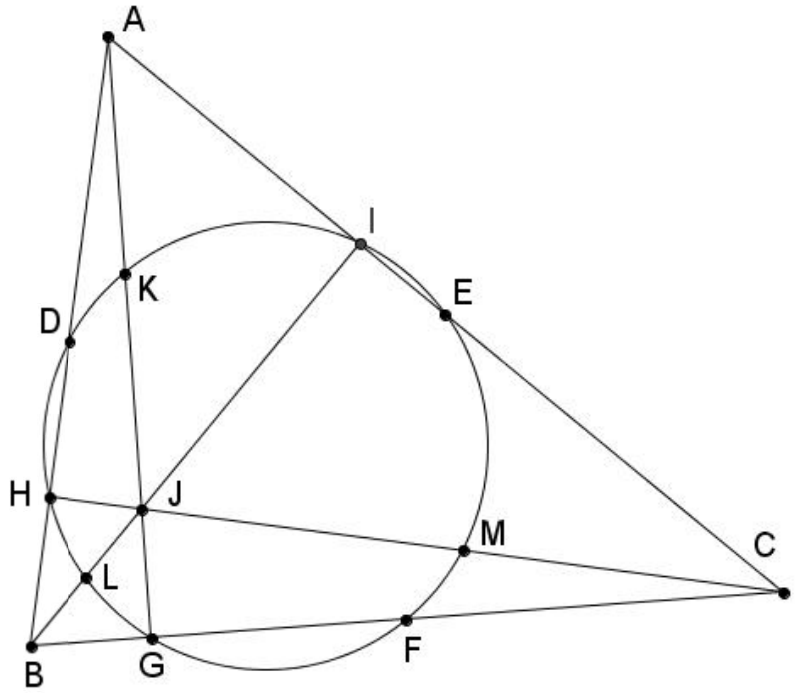
# Overview

1. Illustrative Example
2. A Domain-dependent Approach
3. Implementation and Experiments
4. Conclusions

# Problem

In geometry, the nine-point circle is a circle that can be constructed for any given triangle. It is so named because it passes through nine significant concyclic points defined from the triangle. These nine points are:

- the midpoint of each side of the triangle;
- the foot of each altitude;
- the midpoint of the line segments from vertices of the triangle to the orthocenter (where the three altitudes meet; these line segments lie on their respective altitudes).



# Searching Results by Google

## In terms of "All"

circle triangle midpoint foot

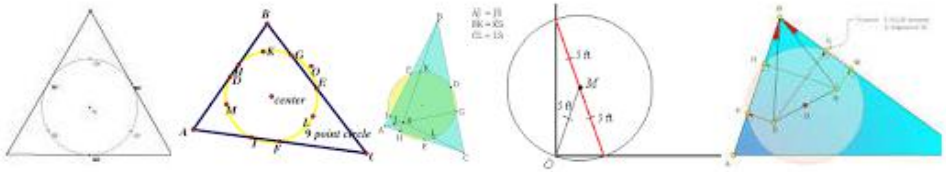
全部 图片 视频 购物 新闻 更多 搜索工具

找到约 528,000 条结果 （用时 0.58 秒）

**Orthocenters and the Nine Point Circle**
jwilson.coe.uga.edu/EMAT6680Fa07/.../assingment8.html ▾ 翻译此页
Construct the nine point circles for each of the four resulting triangles ABC, HBC ... The midpoint of each side of the triangle (3 points); The foot of each altitude (3 ...

**The Nine-Point circle for any triangle passes through the thre...**
jwilson.coe.uga.edu/EMAT6680Fa06/.../Assignment%204.htm ▾ 翻译此页
If the nine point circle is to pass through the midpoints of the sides of the triangle, locate them next, call them D, E, and F. Next we need the three feet of the ...

**circle triangle midpoint foot的图片搜索结果** 举报图片

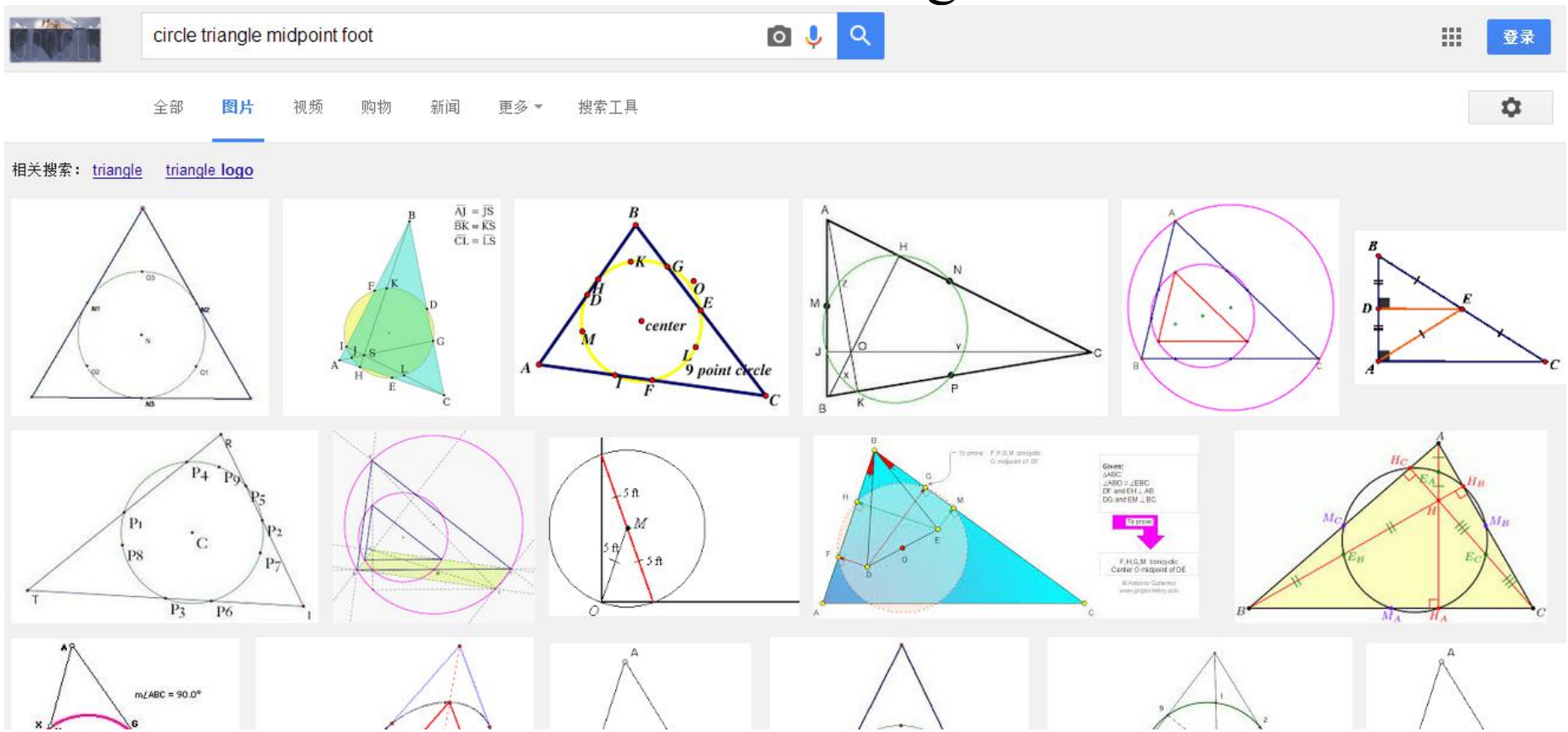


有关"circle triangle midpoint foot"的更多图片

**Nine-point circle - Wikipedia, the free encyclopedia**
https://en.wikipedia.org/wiki/Nine-point_circle ▾ 翻译此页

# Searching Results by Google

In terms of "Images"

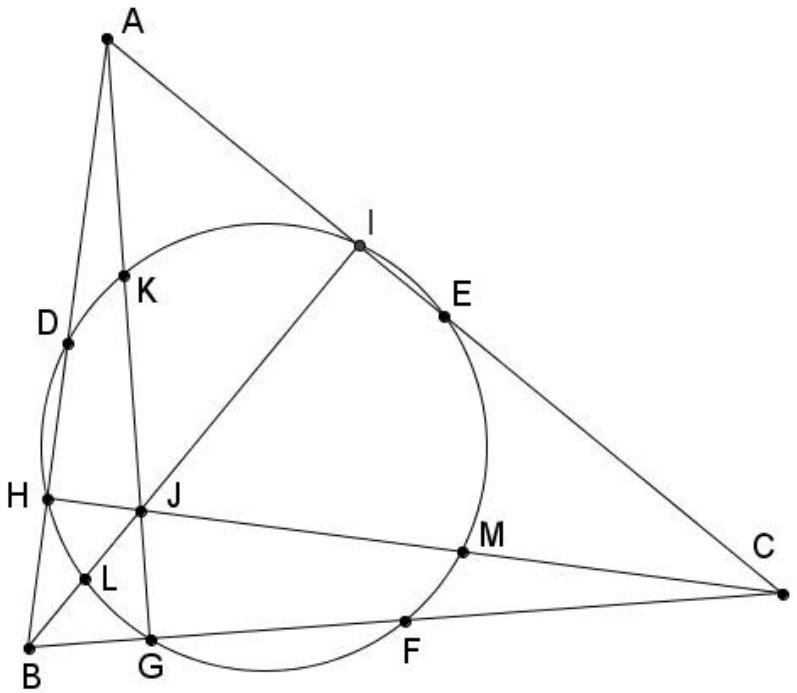



How to generate meaningful geometric keywords from images?

# Some Ideas

We propose to generate geometric keywords from images according to the following steps:

- retrieve basic geometric information;
- derive advanced geometric information;
- generate geometric keywords for searching.



# Retrievie Basic Geometric Information

- Recognized basic geometric entities:

  14 points

  6 lines

  1 circle

- Recognized labels of geometric entities:

  A, B, C, D, E, F, G, H, I, J, K, L, M

# Retrieve Basic Geometric Information (cont.)

- Mined basic geometric relations:

12 incident(A, l)

9 pointOnC(A, o)

0 parallel($l_1$, $l_2$)

3 perpendicular($l_1$, $l_2$)

6 equal(distance(A, B), distance(C, D))

0 equal(size(angle(A, B, C)),
size(angle(D, E, F)))

# Output

**Points**

G := (108,412), C := (484,384), B := (36,417), J := (103,336),
A := ( 82,  58), H := ( 48,329),  I := (231,179), N := (176,299),
F := (261,401), K := ( 93,194), M := (294,360), E := (285,223),
L := ( 68,378),  D := ( 61,231).

**Lines**

a := segment(B,C), b := segment(A,G), c := segment(H,C),
d := segment(A,C), e := segment(B, I),  f := segment(B,A).

**Circle**

g := circle(N,133).

# Output (cont.)

**Relations**

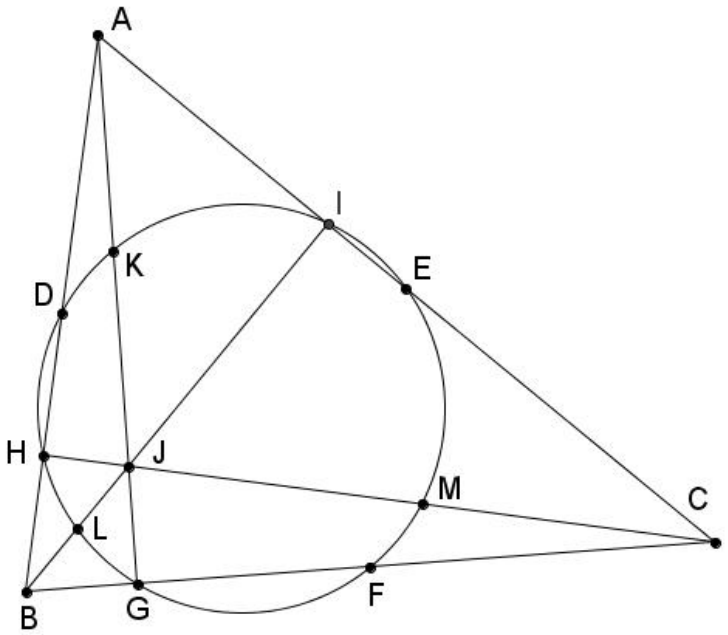
incident(G,a), incident(J,b), incident(J,c), incident(J,e),
incident(H,f), incident(I,d), incident(F,a), incident(K,b),
incident(M,c), incident(E,d), incident(L,e), incident(D,f),
pointOnC(G,g), pointOnC(H,g), pointOnC(I,g),
pointOnC(F,g), pointOnC(K,g), pointOnC(M,g),
pointOnC(E,g), pointOnC(L,g), pointOnC(D,g),
perpendicular(a,b), perpendicular(c,f), perpendicular(d,e),
equal(distance(C,F),distance(B,F)),
equal(distance(J,K),distance(A,K)),
equal(distance(C,M),distance(J,M)),
equal(distance(C,E),distance(A,E)),
equal(distance(B,L),distance(J,L)),
equal(distance(B,D),distance(A,D)).

# Derive Advanced Geometric Information

midpoint, intersection, foot

triangle

concurrent




{D, E, F, K, L, M}, {J}, {G, H, I}

$\Delta ABC$

concurrent(segment(A,G),segment(B,I),segment(C,H))

# Output

## Type of Point

C := (484,384), B := (36,417), A := (82,58), N := (176,299),

F := **midpoint**(C,B), K := **midpoint**(J,A), M := **midpoint**(C,J),
E := **midpoint**(C,A), L := **midpoint**(B,J), D := **midpoint**(B,A),

J := **intersection**(b,c),

H := **foot**(c,segment(B,A)),I := **foot**(segment(A,C),e),
G := **foot**(segment(B,C),b).

## Type of Line

b := halfline(A,G),c := segment(H,C),e := segment(B,I),
h := **triangle**(A,B,C).

## Type of Circle

g := circle(N,133).

# Output (cont.)

**Relations**

**concurrent**(b,c,e),

pointOnC(G,g), pointOnC(H,g), pointOnC(I,g),
pointOnC(F,g), pointOnC(K,g), pointOnC(M,g),
pointOnC(E,g), pointOnC(L,g), pointOnC(D,g).

# Generate Geometric Keywords for Searching

Proper strategies are required to generate keywords for web searching, i.e., determining the sequence of classes of geometric information for keywords generation.

Keywords are weighted according to the levels and types of their corresponding geometric entities or relations (e.g., advanced >basic, entities >relations).

It is possible to design specific strategies according to one's own requirements.

# Output

For the given image, four groups of keywords are generated:

1. {triangle, circle, closest, concurrent, midpoint}

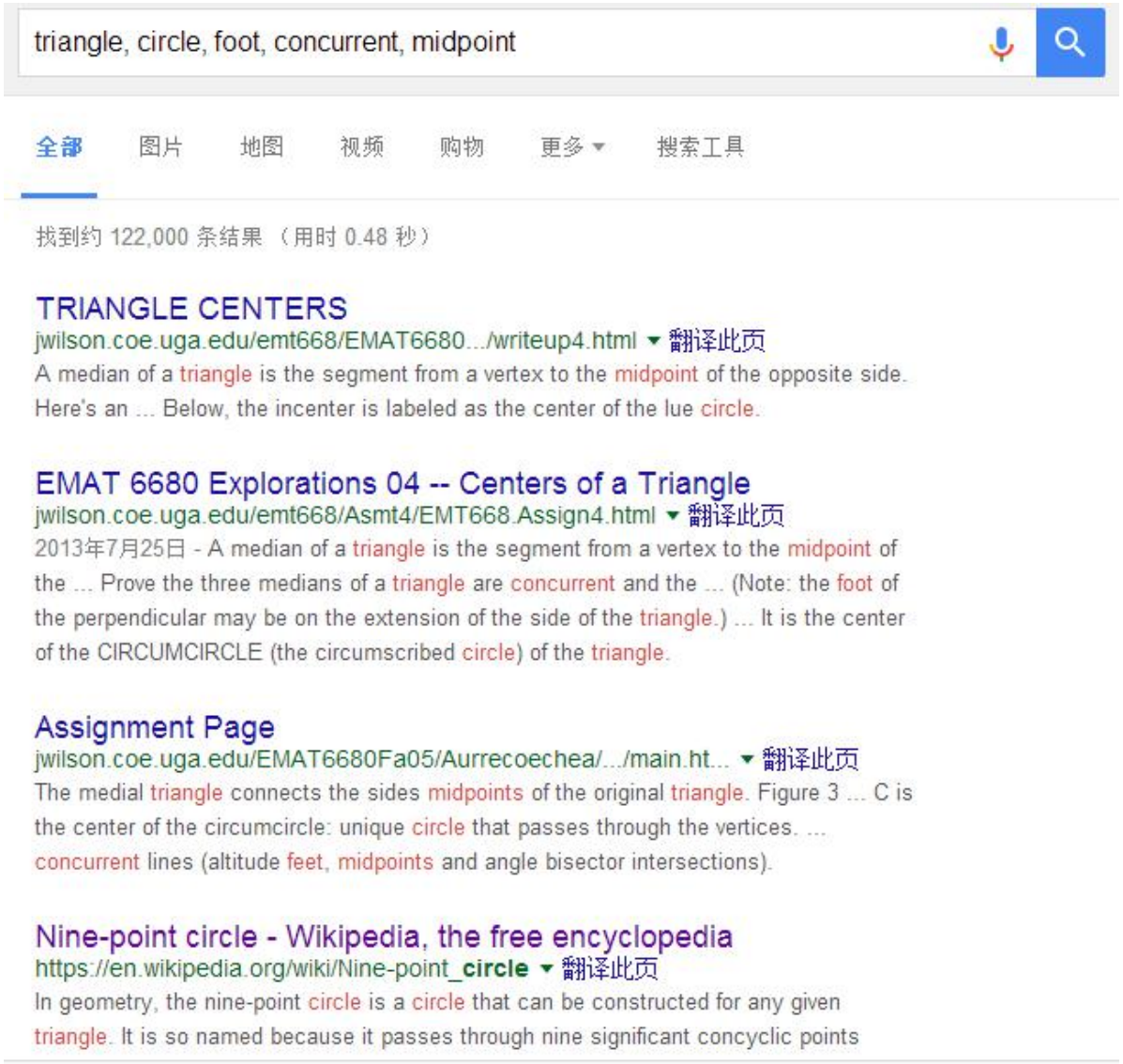
2. **{triangle, circle, foot, concurrent, midpoint}**

3. {triangle, circle, closest, concurrent, bisect}

4. {triangle, circle, foot, concurrent, bisect}

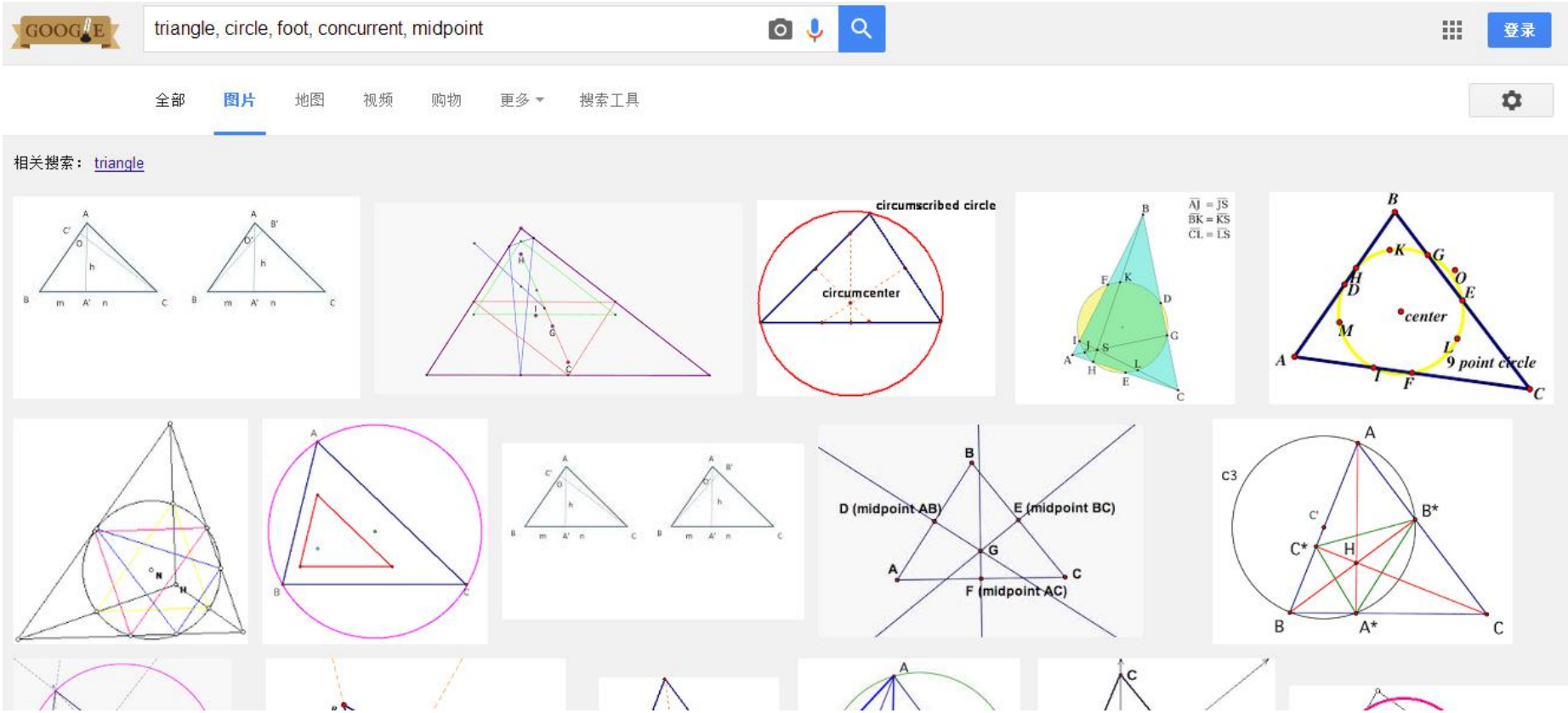
# Search by Google

## In terms of "All"



# Search by Google

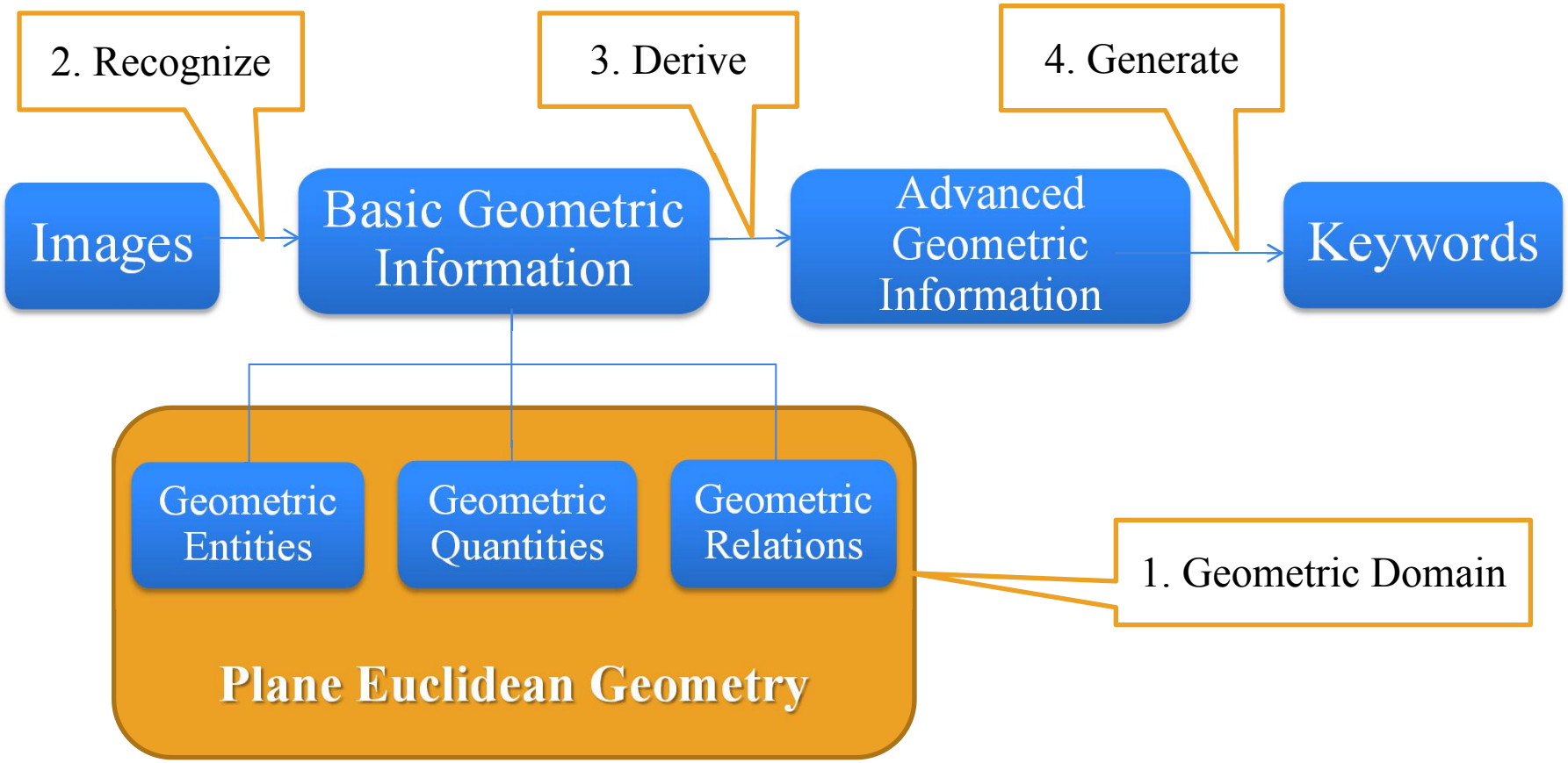
## In terms of "Images"



# Overview

1. Illustrative Example
2. A Domain-dependent Approach
3. Implementation and Experiments
4. Conclusions

# Overview of the Domain-dependent Approach



# Specify the Domain of Interest

- Identify a set of *basic geometric concepts*, including entities, quantities, and relations, and a set of *advanced* ones with formal representation.

# Specify the Domain of Interest (cont.)

## Basic Geometric Entities

| Type | Representation | Meaning | Keywords |
|---|---|---|---|
| Point | (x,y) | a point with coordinates (x,y) | point |
| Line | line(A,B) *or* halfline(A,B) *or* segment(A,B) | a straight line *or* a halfline *or* a segment passing through two different points A and B | line *or* halfline *or* segment |
| Circle | circle(A,r) *or* circle(A,B) *or* circle(A,B,C) | a circle with center A and radius r *or* a circle with center A and passing through another point B *or* a circle passing through three different points A, B, C | circle |

# Specify the Domain of Interest (cont.)

## Basic Geometric Quantities

| Type | Representation | Meaning | Keywords |
|---|---|---|---|
| Distance | distance(A,B) | the distance between A and B where A and B are two points | distance |
| Angle | angle(A,B,C) | ∠ABC where A, B, and C are three different points | angle |
| Size | size( α ) | the size of α where α is an angle | size |

# Specify the Domain of Interest (cont.)

## Basic Geometric Relations

| Type | Representation | Meaning | Keywords |
|---|---|---|---|
| Boolean | incident(A,l) | a point A lies on a line l | collinear, incident |
| Boolean | pointOnC(A,o) | a point A is on a circle o | incident |
| Boolean | parallel($l_1$,$l_2$) | a line $l_1$ is parallel to a line $l_2$ | parallel |
| Boolean | perpendicular($l_1$,$l_2$) | a line $l_1$ is perpendicular to a line $l_2$ | perpendicular |
| Boolean | equal(distance(A, B),distance(C,D)) | the distance between two points A and B is equal to the distance between two points C and D | equidistant |
| Boolean | equal( size(angle(A,B,C)), size(angle(D,E,F))) | the size of ∠ABC is equal to the size of ∠DEF | equal angle |

# Specify the Domain of Interest (cont.)

## Advanced Geometric Entities

| Name | Representation | Definition | Keywords |
|---|---|---|---|
| midpoint | M := midpoint(A,B) | midpoint(A::Point, B::Point) ≜ [M::Point where incident(M, line(A,B)) ∧ equal(distance(M, A),distance(M, B))] | midpoint, bisect |
| Triangle | t := triangle(A,B,C) | triangle(A::Point,B::Point,C::Point) ≜ [{segment(A,B), segment(B,C),segment(C,A)} where ¬incident(A,segment(B,C))] | triangle |
| Circumcircle | c := circumcircle(t) | c := circumcircle(t::Triangle) ≜ [circumcircle(O::Point, r::Length) where t := triangle(A,B,C) ∧ pointOnC(A,c) ∧ pointOnC(B,c) ∧ pointOnC(C,c)] | circumcircle |
| ... | ... | ... | ... |

# Specify the Geometric Domain (cont.)

## Advanced Geometric Relations

| Name | Representation | Definition | Keywords |
|---|---|---|---|
| Tangent | tangent(l,c) | tangent(l::Line,c::Circle) ≜ [pointOnC(foot(center(c),l),c)] | tangent |
| Bisect | bisect(A,B,C,D) | bisect(A::Point,B::Point,C::Point,D::Point) ≜ [equal(size(angle(A,B,D)), size(angle(C,B,D)))] | angle bisector |
| Trisect | trisect(A,B,C,D) | trisect(A::Point,B::Point,C::Point,D::Point, E::Point) ≜ [equal(size(angle(A,B,D)), size(angle(D,B,E))) ∧ equal(size(angle(D,B,E)), size(angle(E,B,C)))] | trisector, trisect |
| Concurrent | concurrent($l_1$,$l_2$,$l_3$) | concurrent(l1::Line,l2::Line,l3::Line) ≜ [incident(intersection(l1,l2),l3) ∨ incident(intersection(l2,l3),l1) ∨ incident(intersection(l1,l3),l2)] | concurrent |

# Retrieve Basic Geometric Information

Use *shape recognition techniques* (e.g., improved Hough transform, randomized detection algorithm) to detect *geometric entities* from the given image.

Use techniques of *image matching* (e.g., OCR Engines) to recognize *labels* of the detected entities.

Use *numerical computation* to mine *geometric relations* among the detected entities.

# Derive Advanced Geometric Information

For each geometric instance O, we introduce a property level (denoted as O.level), which is an integer in [0,+∞), to characterize the priority of O in the derivation process.

Derive advanced geometric entities and relations in order from lower level to higher level according to their definitions.

# Generate Keywords

- Keywords of different levels are weighted for their influences on the characterization of the image, and
- these keywords for geometric concepts are combined to generate several groups of keywords ordered according to their weights.

# Overview

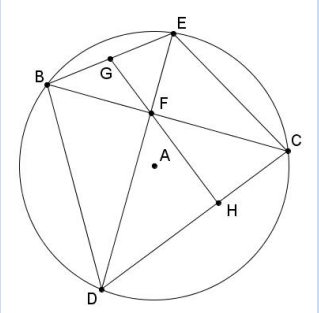
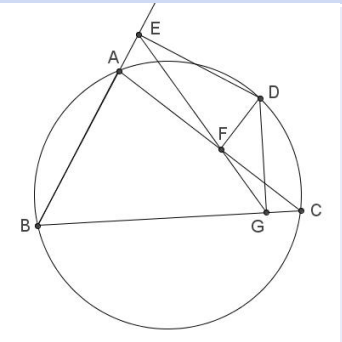
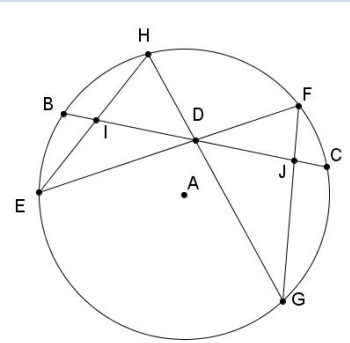
1. Illustrative Example
2. A Domain-dependent Approach
3. Implementation and Experiments
4. Conclusions

# Implementation

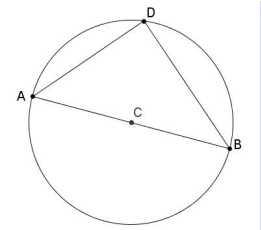
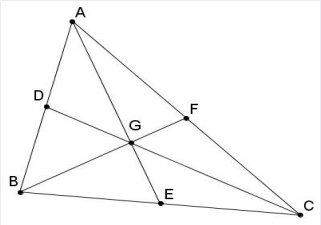
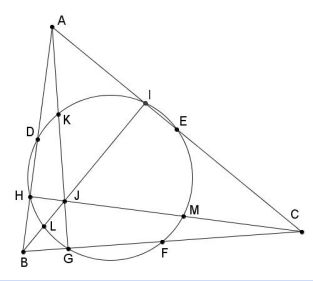
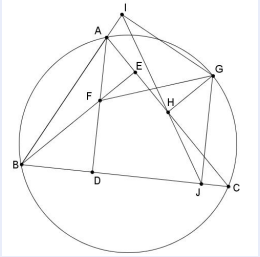
- The techniques of keyword generation from an image of diagram described above have been implemented in C++ development environment.

- Diagram images used for our experiments were produced by using GeoGebra.

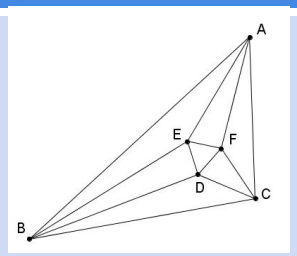
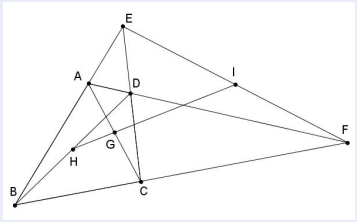
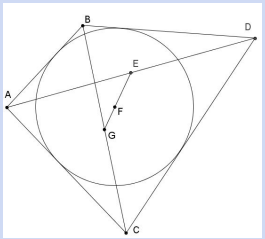
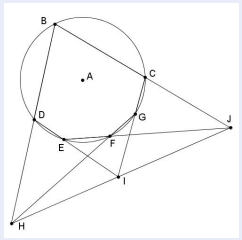
# Experiments

| No. | Image | Keywords |
|---|---|---|
| 1 |   | {quadrilateral, circumcircle, closest/foot, midpoint/bisect} |
| 2 |   | {triangle, circumcircle, closest/foot, collinear/incident} |
| 3 |   | {segment, circle, midpoint/bisect, equidistant} |

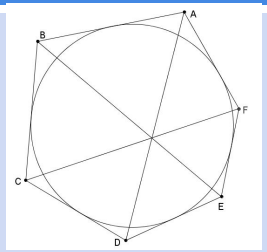
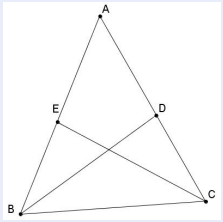
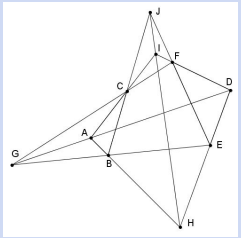
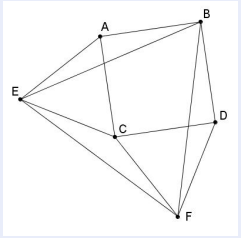
# Experiments (cont.)

| No. | Image | Keywords |
|---|---|---|
| 4 |   | {triangle, circumcircle, midpoint/bisect, perpendicular} |
| 5 |   | {triangle, midpoint/bisect, concurrent, intersection/intersect} |
| 6 |   | {triangle, circle, closest/foot, concurrent, midpoint/bisect} |
| 7 |   | {triangle, circumcircle, midpoint/bisect, perpendicular} |

# Experiments (cont.)

| No. | Image | Keywords |
|---|---|---|
| 8 |  | {triangle, trisector, equidistant} |
| 9 |  | {triangle, midpoint/bisect, collinear/ incident, intersection/intersect} |
| 10 |  | {quadrilateral, incircle, midpoint/ bisect, collinear/incident} |
| 11 |  | {polygon, circumcircle, intersection /intersect, concurrent} |

# Experiments (cont.)

| No. | Image | Keywords |
|---|---|---|
| 12 |  | {polygon, incircle, concurrent} |
| 13 |  | {triangle, intersection, angle bisector, equidistant} |
| 14 |  | {polygon, intersection, collinear/incident, triangle} |
| 15 |  | {polygon, intersection, perpendicular, equidistant} |

# Overview of Experiments

1. The search engine used in our experiments is Google.

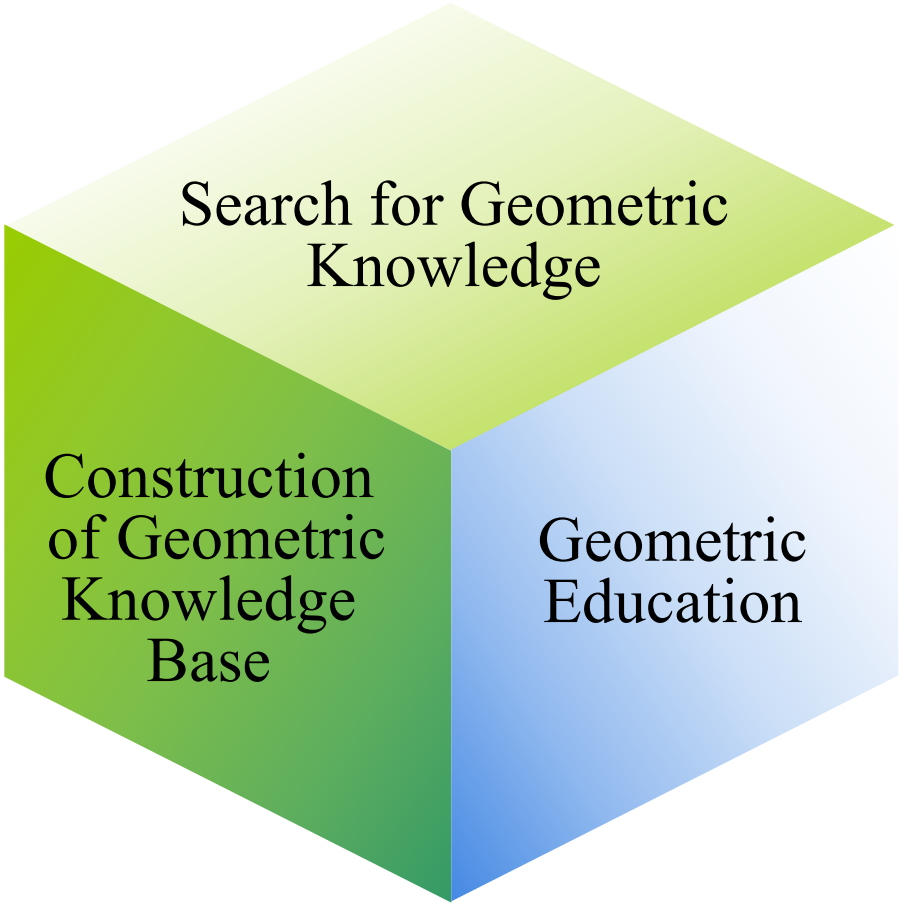
2. Among 40 examples, more than 50% are encouraging, which indicates that the keywords generated are effective for searching.

3. More experimental results will be performed in our future work.

# Overview

1. Illustrative Example
2. A Domain-dependent Approach
3. Implementation and Experiments
4. Conclusions

# Applications



# Conclusions

1. The presented approach can be used to search the web for geometric information, in particular geometric theorems, which a query image may imply.

2. The approach may be generalized for the generation of keywords in any other domain of interest from images in the domain.

3. The approach has a potential application in automated label generation for images.

# Future Work

We are extending our work towards

1. discovering of mathematical concepts from image data, and

2. automating the process of digitalizing the classic mathematical literature with semantic representation.

# Automated Generation of Keywords from Images for Geometric Information Search

Thanks!