Dilemma about image cropping algorithm - is it possible?

Asked 16 years, 2 months ago Modified 15 years, 9 months ago

Viewed 3k times

6

I am building a web application using .NET 3.5 (ASP.NET, SQL Server, C#, WCF, WF, etc) and I have run into a major design dilemma. This is a uni project btw, but it is 100% up to me what I develop.

I need to design a system whereby I can take an image and automatically crop a certain object within it, without user input. So for example, cut out the car in a picture of a road. I've given this a lot of thought, and I can't see any feasible method. I guess this thread is to discuss the issues and feasibility of achieving this goal. Eventually, I would get the dimensions of a car (or whatever it may be), and then pass this into a 3d modelling app (custom) as parameters, to render a 3d model. This last step is a lot more feasible. It's the cropping issue which is an issue. I have thought of all sorts of ideas, like getting the colour of the car and then the outline around that colour. So if the car (example) is yellow, when there is a yellow pixel in the image, trace around it. But this would fail if there are two yellow cars in a photo.

Ideally, I would like the system to be completely automated. But I guess I can't have everything my way.

Also, my skills are in what I mentioned above (.NET 3.5, SQL Server, AJAX, web design) as opposed to C++ but I would be open to any solution just to see the feasibility.

I also found this patent: US Patent 7034848 - System and method for automatically cropping graphical images

Thanks

algorithm image-processing computer-vision

Share

Improve this question

Follow

edited Feb 27, 2009 at 17:03

Eddie

54.4k ● 22 ● 128 ● 146

asked Oct 18, 2008 at 12:41

GSS

157 ● 3 ● 4

Just do your best at tagging. As you can see, the community will correct you. :) – Bill the Lizard Oct 18, 2008 at 23:34

8 Answers

Sorted by: Highest score (default)

2

This is one of the problems that needed to be solved to finish the DARPA Grand Challenge. Google video has a great presentation by the project lead from the winning team, where he talks about how they went about their solution, and how some of the other teams approached it. The relevant portion starts around 19:30 of the video, but it's a great talk, and the whole thing is worth a watch. Hopefully it gives you a good starting point for solving your problem.

Share Improve this answer Follow

edited Oct 18, 2008 at 23:56

answered Oct 18, 2008 at 23:40

Bill the Lizard

405k ● 211 ● 572 ● 889

2

What you are talking about is an open research problem, or even several research problems. One way to tackle this, is by image segmentation. If you can safely assume that there is one object of interest in the image, you can try a figure-ground segmentation algorithm. There are many such algorithms, and none of them are perfect. They usually output a segmentation mask: a binary image where the figure is white and the background is black. You would then find the bounding box of the figure, and use it to crop. The thing to remember is that none of the existing segmentation algorithm will give you what you want 100% of the time.

Alternatively, if you know ahead of time what specific type of object you need to crop (car, person, motorcycle), then you can try an object detection algorithm. Once again, there are many, and none of them are perfect either. On the other hand, some of them may work better than segmentation if your object of interest is on very cluttered background.

To summarize, if you wish to pursue this, you would have to read a fair number of computer vision papers, and try a fair number of different algorithms. You will also increase your chances of success if you constrain your problem domain as much as possible: for example restrict yourself to a small number of object categories, assume there is only one object of interest in an image, or restrict yourself to a certain type of scenes (nature, sea, etc.). Also keep in mind, that even the accuracy of state-of-the-art approaches to solving this type of problems has a lot of room for improvement.

And by the way, the choice of language or platform for this project is by far the least difficult part.

edited Oct 19, 2008 at 2:30

answered Oct 19, 2008 at 2:23

Dima
39.3k ● 14 ● 78 ● 116

2

A method often used for face detection in images is through the use of a Haar classifier cascade. A classifier cascade can be trained to detect any objects, not just faces, but the ability of the classifier is highly dependent on the quality of the training data.

This paper by Viola and Jones explains how it works and how it can be optimised.

Although it is C++ you might want to take a look at the image processing libraries provided by the OpenCV project which include code to both train and use Haar cascades. You will need a set of car and non-car images to train a system!

Share Improve this answer

Follow

answered Oct 22, 2008 at 16:50

Dan

688 ● 4 ● 9

0

Some of the best attempts I've see of this is using a large database of images to help understand the image you have. These days you have flickr, which is not only a giant corpus of images, but it's also tagged with meta-information about what the image is.

Some projects that do this are documented here:

http://blogs.zdnet.com/emergingtech/?p=629

Share Improve this answer

answered Oct 18, 2008 at 13:12

Follow

Lou Franco
89k • 14 • 136 • 198

0

Start with analyzing the images yourself. That way you can formulate the criteria on which to match the car. And you get to define what you cannot match.

If all cars have the same background, for example, it need not be that complex. But your example states a car on a street. There may be parked cars. Should they be recognized?

If you have access to MatLab, you could test your pattern recognition filters with specialized software like PRTools.

Wwhen I was studying (a long time ago:) I used Khoros Cantata and found that an edge filter can simplify the image greatly.

But again, first define the conditions on the input. If you don't do that you will not succeed because pattern recognition is really hard (think about how long it took to crack captcha's)

Share Improve this answer

Follow

answered Oct 18, 2008 at 13:31

extraneon
23.9k • 2 • 49 • 51

I did say photo, so this could be a black car with a black background. I did think of specifying the colour of the object, and then when that colour is found, trace around it

0

(high level explanation). But, with a black object in a black background (no constrast in other words), it would be a very difficult task.

Better still, I've come across several sites with 3d models of cars. I could always use this, stick it into a 3d model, and render it.

A 3D model would be easier to work with, a real world photo much harder. It does suck :(

Share Improve this answer

Follow

answered Oct 18, 2008 at 15:36

GSS

-1

If I'm reading this right... This is where AI shines.

I think the "simplest" solution would be to use a neural-network based image recognition algorithm. Unless you know that the car will look the *exact* same in each picture, then that's pretty much the only way.

If it IS the exact same, then you can just search for the pixel pattern, and get the bounding rectangle, and just set the image border to the inner boundary of the rectangle.

Share Improve this answer

Follow

answered Oct 18, 2008 at 12:53

TraumaPony

10.8k ● 13 ● 57 ● 75

-1

I think that you will never get good results without a real user telling the program what to do. Think of it this way: how should your program decide when there is more than 1 *interesting* object present (for example: 2 cars)? what if the object you want is actually the mountain in the background? what if nothing of interest is inside the picture, thus nothing to select as the object to crop out? etc, etc...

With that said, if you can make *assumptions* like: only 1 object will be present, then you can have a go with using image recognition algorithms.
Now that I think of it. I recently got a lecture about artificial intelligence in robots and in robotic research techniques. Their research went on about language interaction, evolution, and language recognition. But in order to do that they also needed some simple image recognition algorithms to process the perceived environment. One of the tricks they used was to make a 3D plot of the image where x and y where the normal x and y axis and the z axis was the brightness of that particular point, then they used the same technique for red-green values, and blue-yellow. And lo and behold they had something (relatively) easy they could use to pick out the objects from the perceived environment. (I'm terribly sorry, but I can't find a link to the nice charts they had that showed how it all worked).

Anyway, the point is that they were not interested (that much) in image recognition so they created something

that worked *good enough* and used something less advanced and thus less time consuming, so it is possible to create something simple for this complex task.

Also any good image editing program has some kind of *magic wand* that will select, with the right amount of tweaking, the object of interest you point it on, maybe it's worth your time to look into that as well.

So, it basically will mean that you:

- have to make some assumptions, otherwise it will fail terribly
- will probably best be served with techniques from AI, and more specifically image recognition
- can take a look at paint.NET and their algorithm for their *magic wand*
- try to use the fact that a *good* photo will have the object of interest somewhere in the middle of the image

.. but i'm not saying that this is *the* solution for your problem, maybe something simpler can be used.

Oh, and I will continue to look for those links, they hold some really valuable information about this topic, but I can't promise anything.

answered Oct 18, 2008 at 13:28

sven

18.3k ● 10 ● 52 ● 62