Interactive Vision

Two methods for Interactive Edge detection.

Final Project by Daniel Zatulovsky

Email: zatulovs@post.bgu.ac.il

Interactive Computer Vision

The purpose of this project is to present a new research field, Interactive Computer Vision, convince the reader that such field is indeed useful and then discuss a specific problem in the field: Interactive edge detection, providing two distinct methods to approach it, each applicable under different situations.

Interactive Computer Vision

Definition: Interactive Computer Vision is the field that tries to solve the Vision problem by getting limited help from a user and by so managing to greatly improve the results.

Applications

There are numerous task that are hard to solve by fully automatic systems. Instead of trying to do so, we'll try to build a system that would require minimal user interaction, thus save effort and resources.

For example, let us think of the CROPS project.

Interactive Edge Detection

Application:

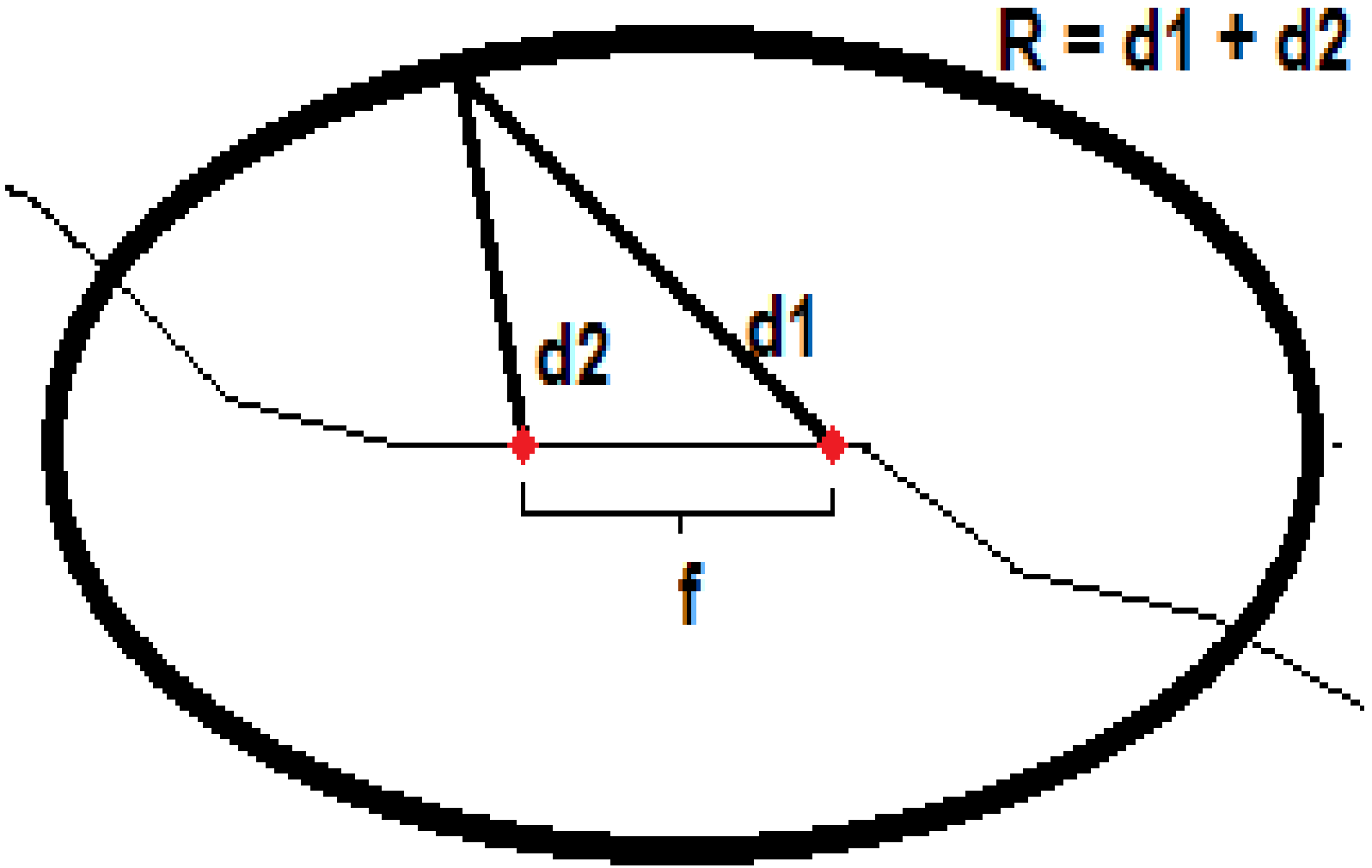
A good example where algorithms such as these would be useful are mobile photo-related applications. A user could, inaccurately, mark the borders of an object and the application would use the input to recognize and process it. For example, you could accurately select and delete a person passing in front of the camera while the photo was taken (of course filling the blank would require a different, unrelated, algorithm).

Proximity based method

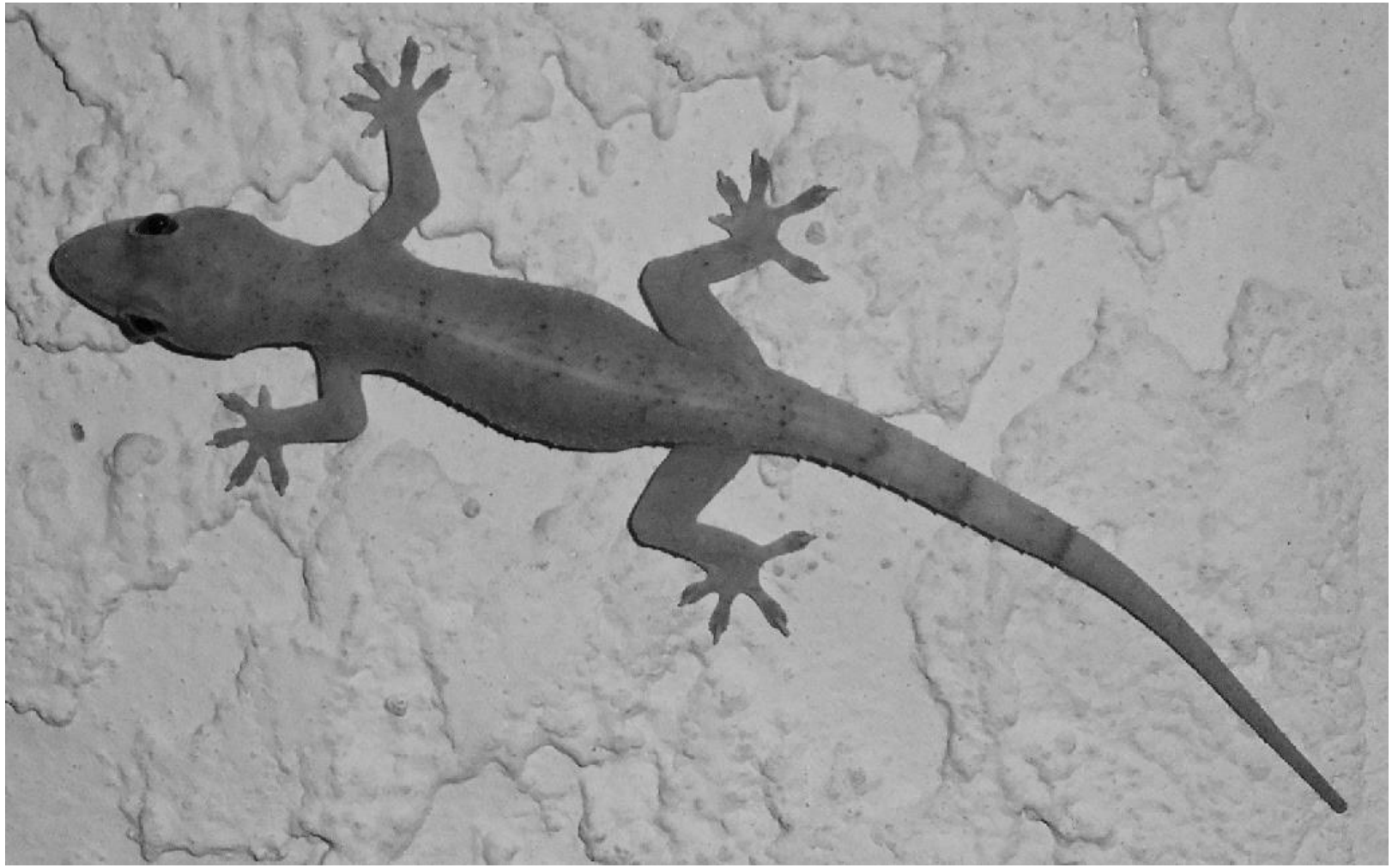
The first, proximity based, method requires the user to trace a curve around the object. The algorithm then decides whether an edge point should be kept or filtered based on its proximity to the curve.

Proximity

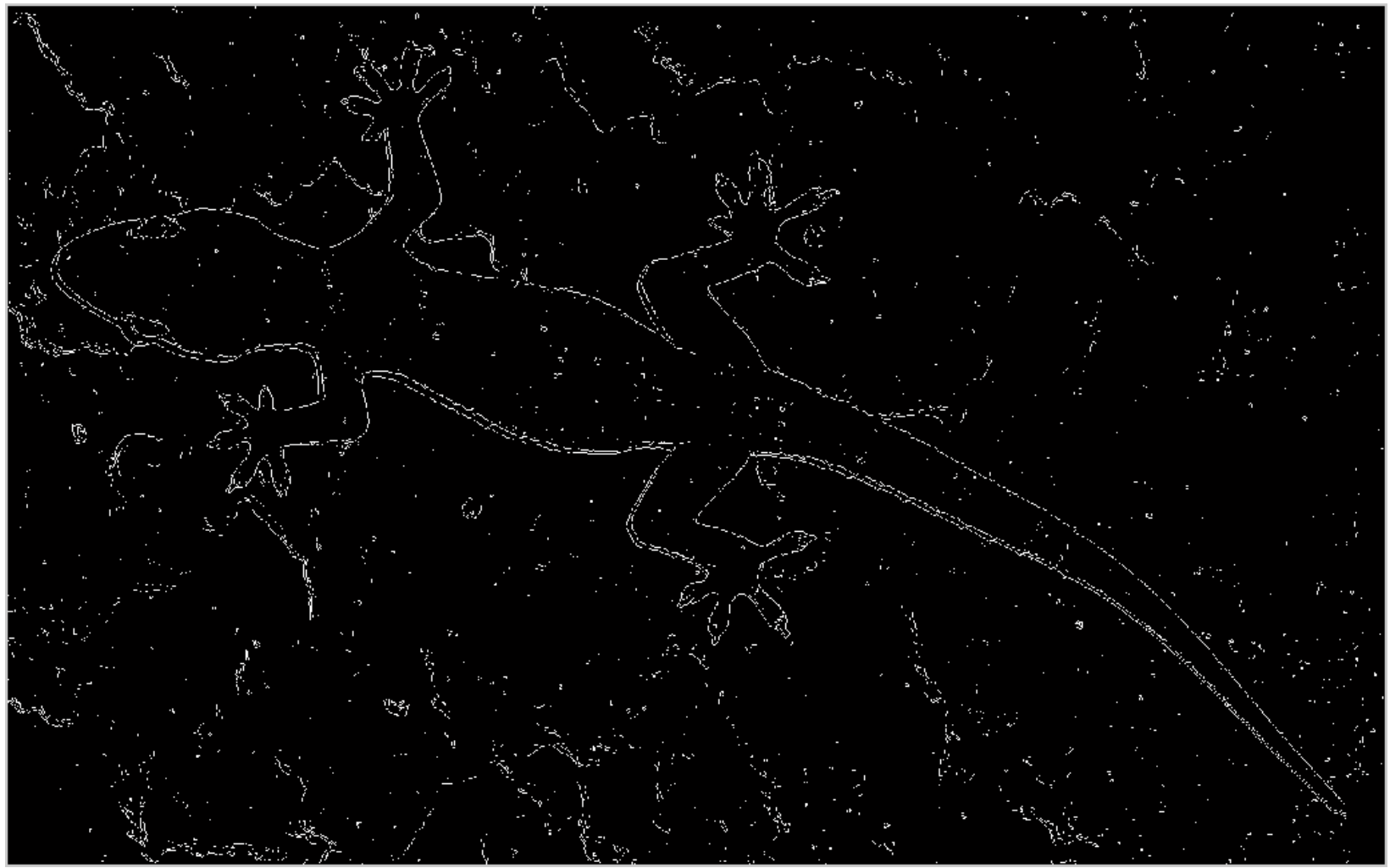
For every point P of the input curve, an elliptic shape is built. The ellipse is centered around P, has the same approximate direction as the curve at P, has the focal distance *f* and the radius *R*. We then filter the original edge map of the image based on those ellipses.



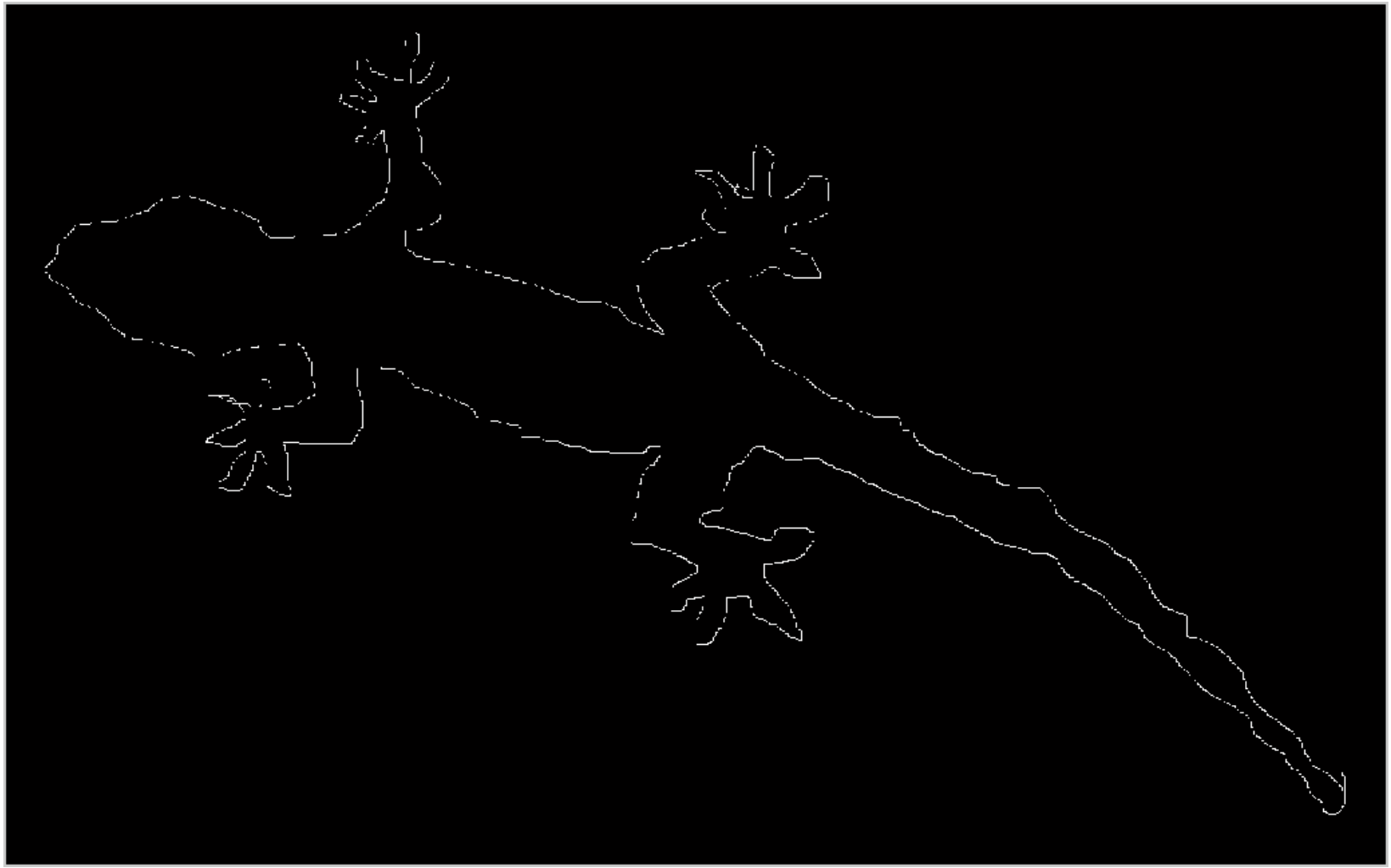
Results



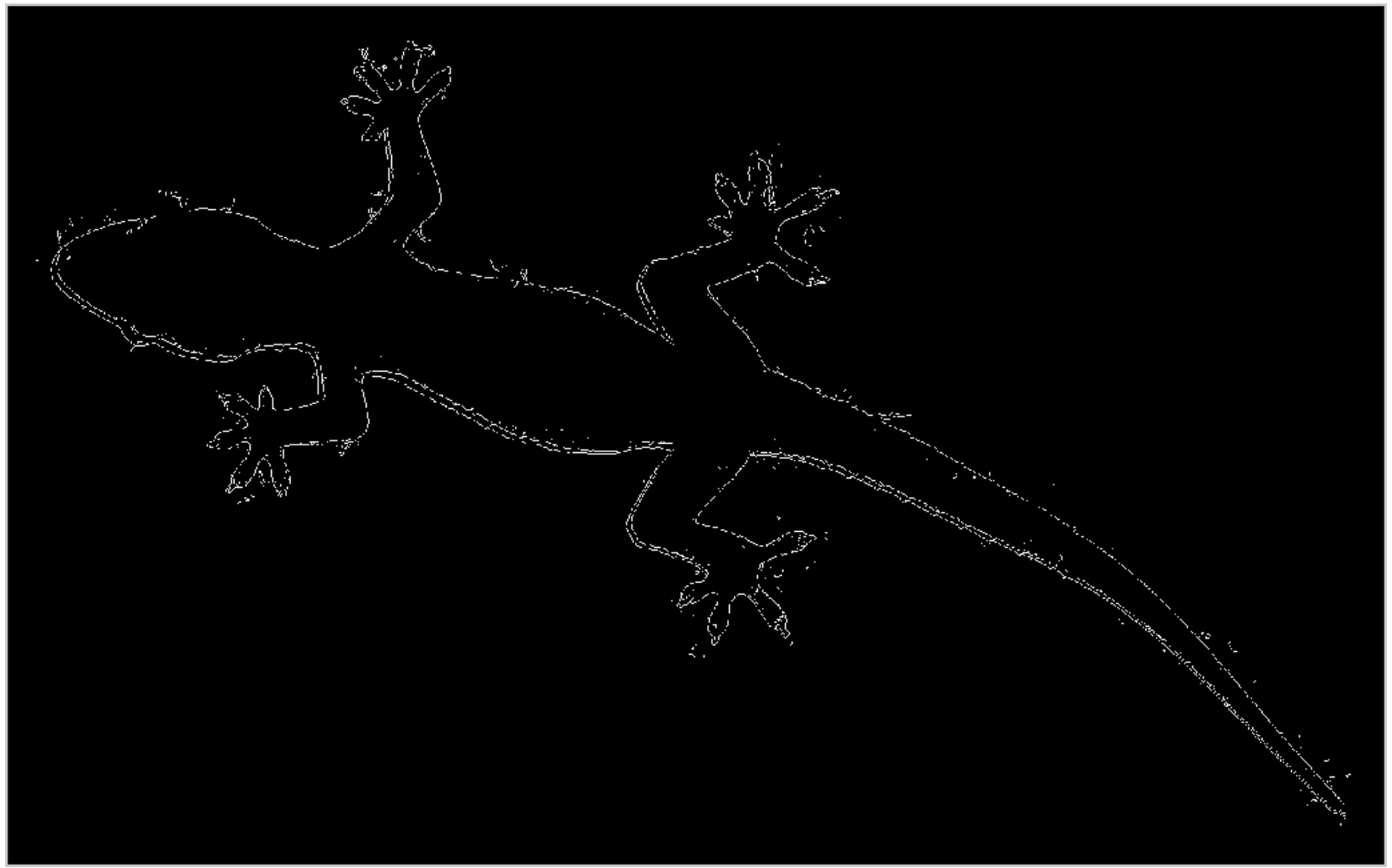
Results



Results

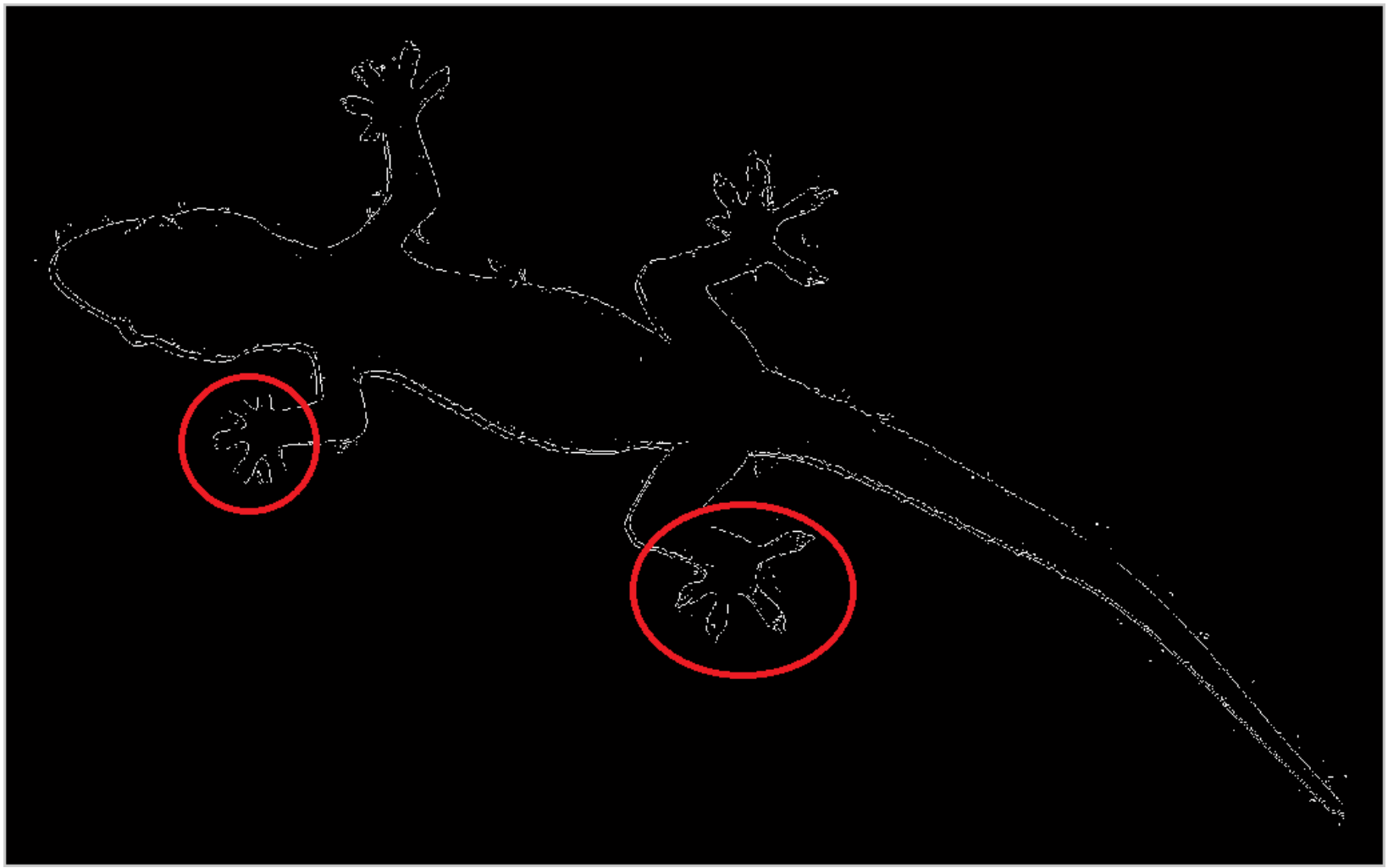


Results



Results

Results



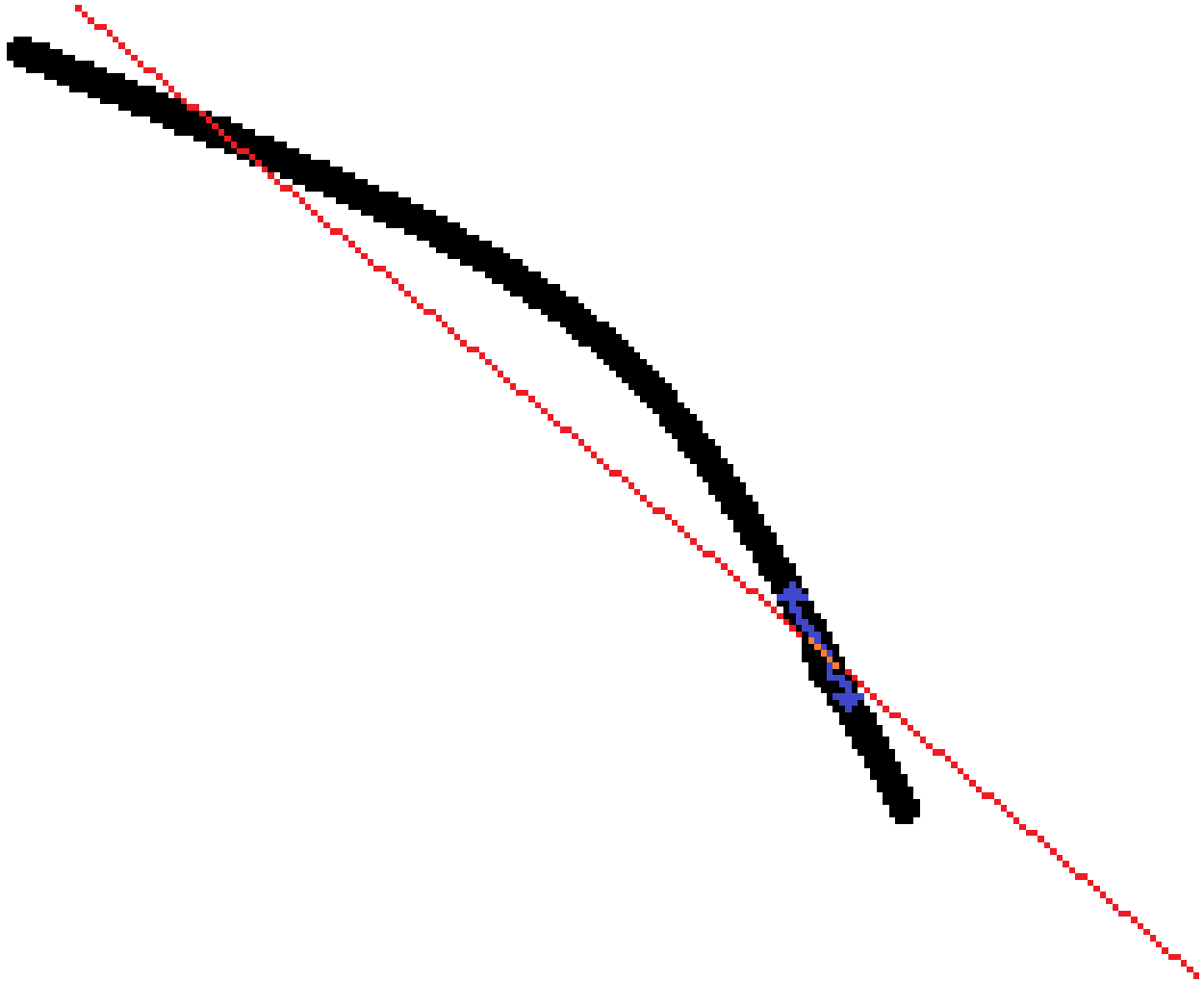
Intersection based method

The second method requires less precise drawing and uses only the intersection points between the user input and the edge map instead. This allows the user to use straight lines instead of curves.

Intersection

Points in the intersection are marked as edges. Then a process similar to Canny's Hysteresis is used to mark all their adjacent points as edges recursively.

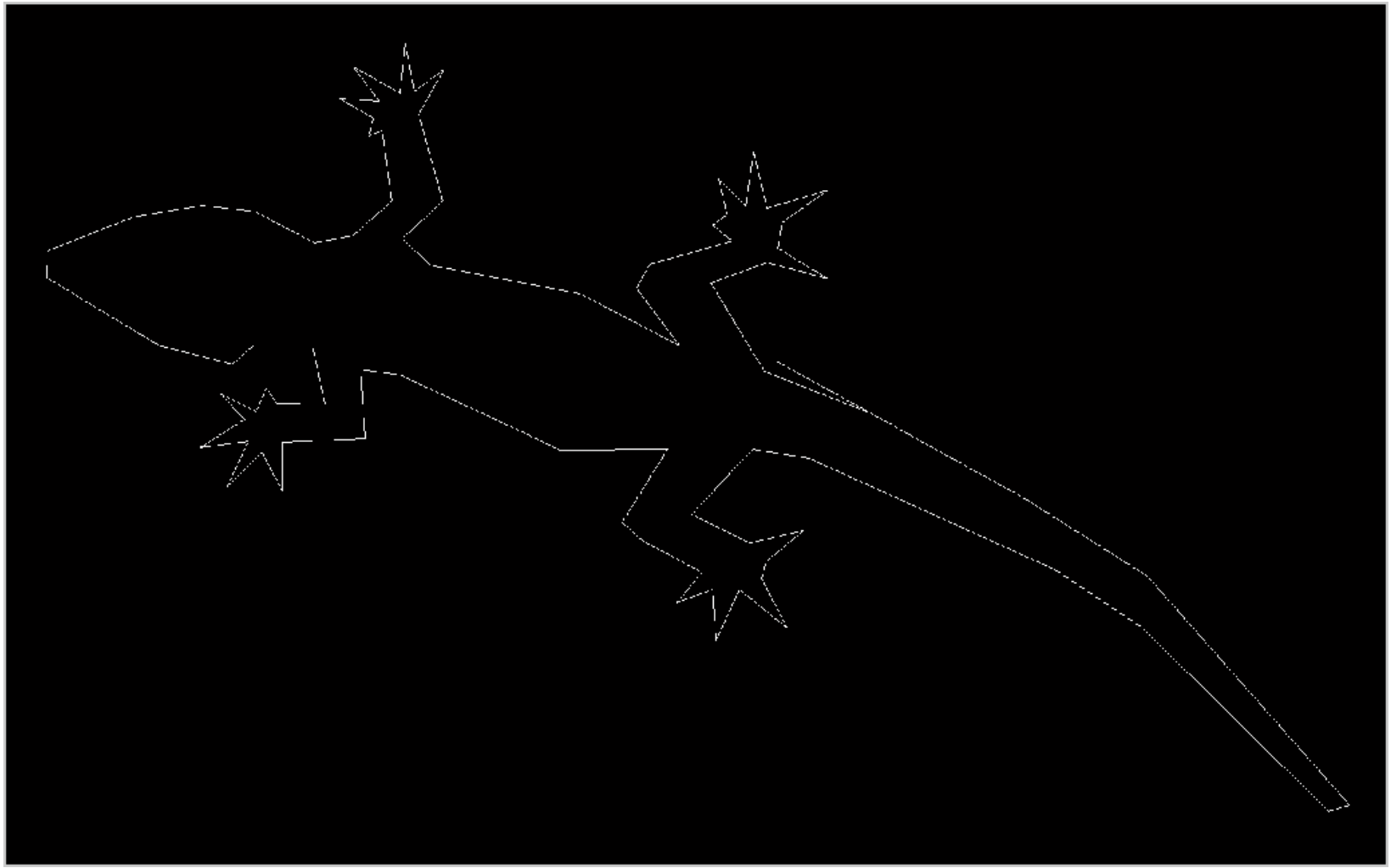
A window centered around each pixel marked as edge is considered and marked as edge and all the pixels in it are marked as edges.



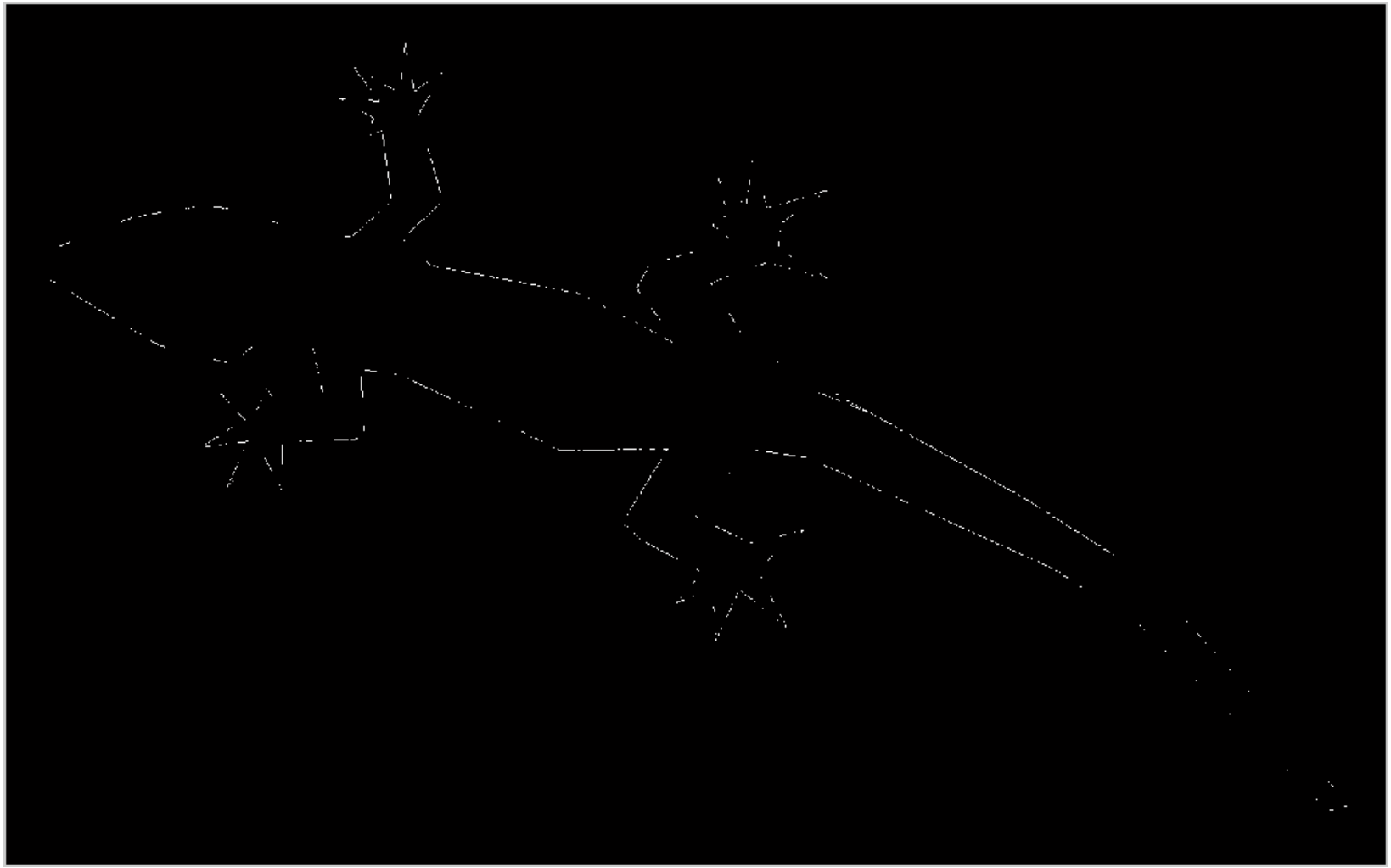
Results

Notice how bright the edges are. This will be important latter.

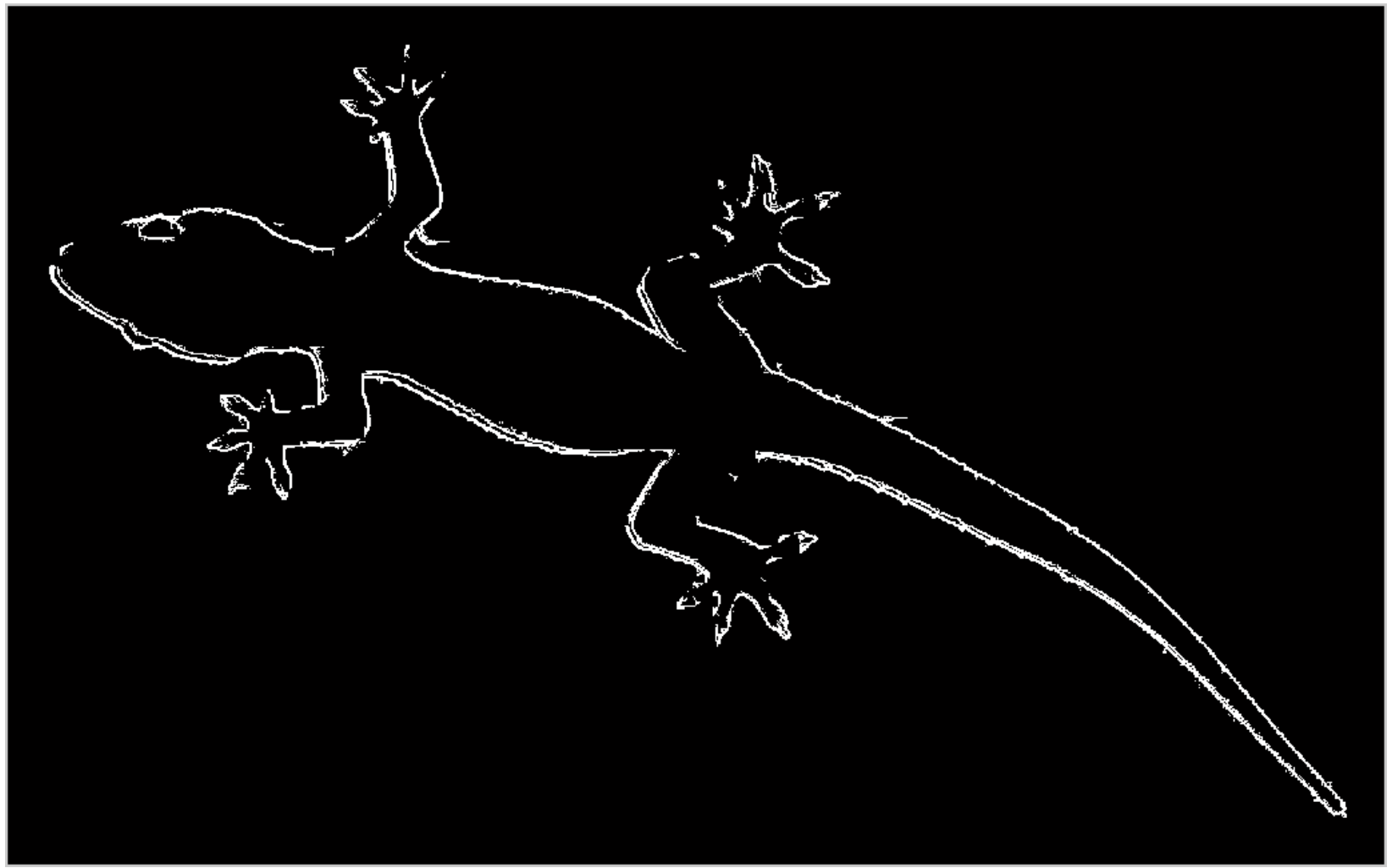
Results



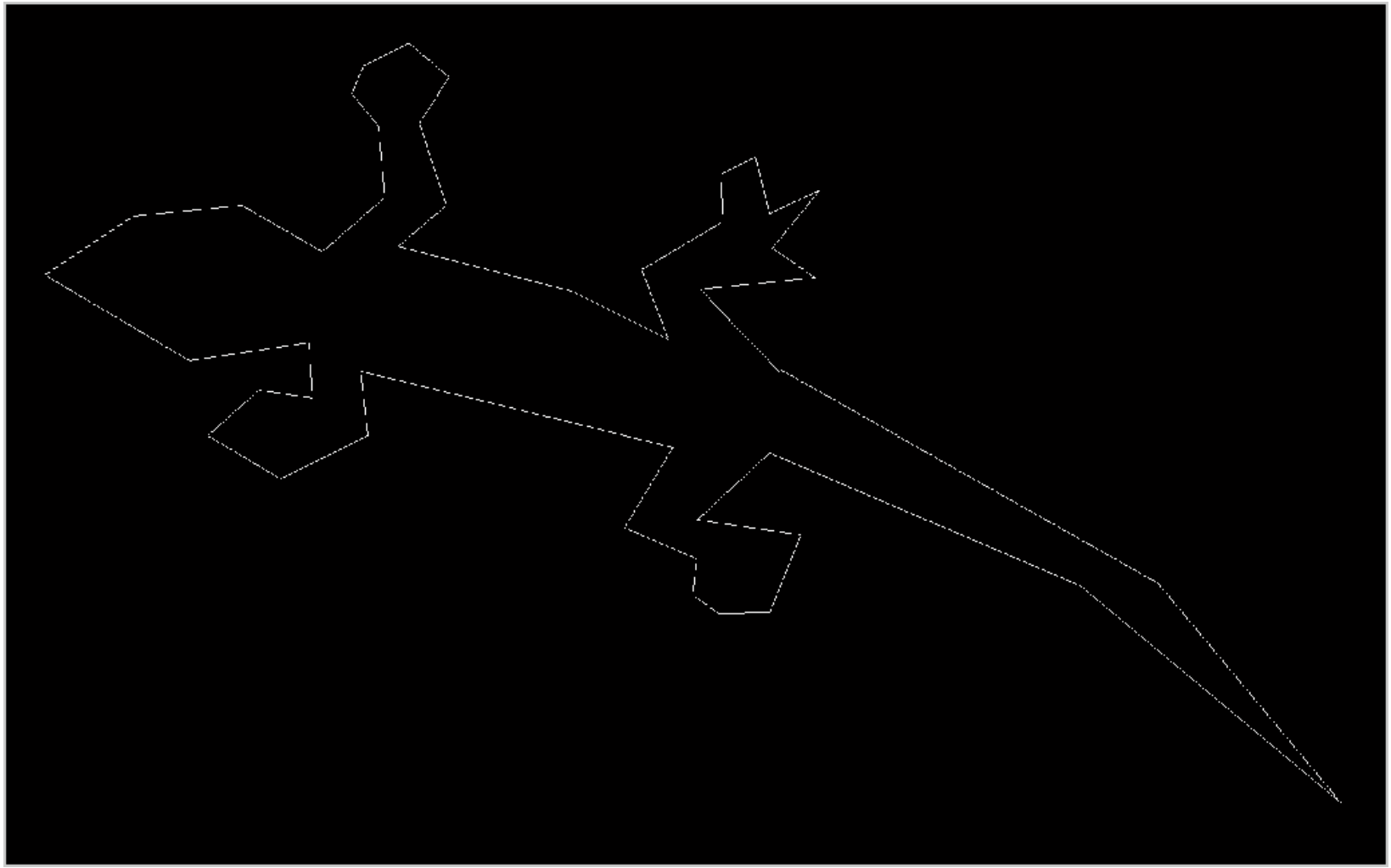
Results



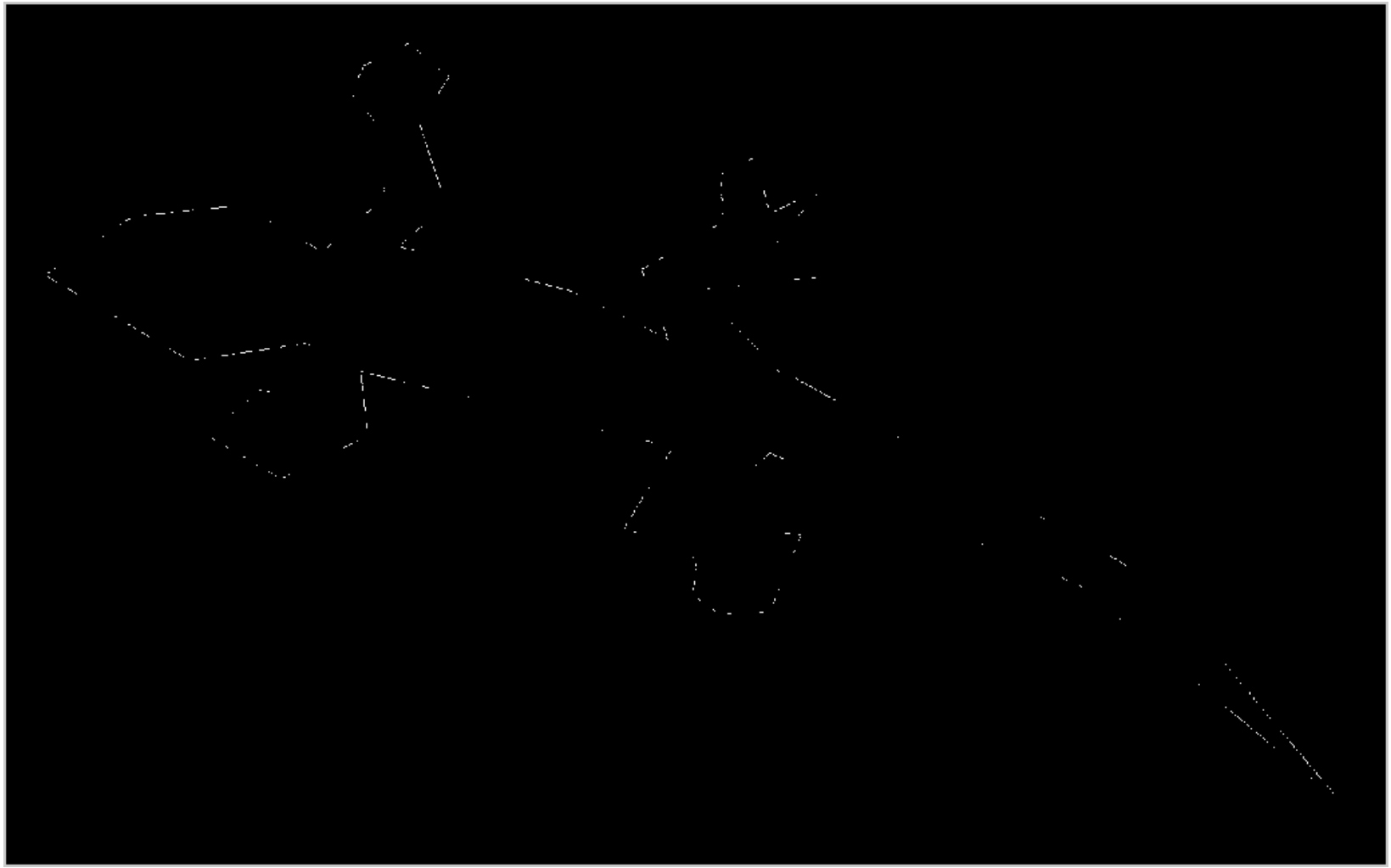
Results



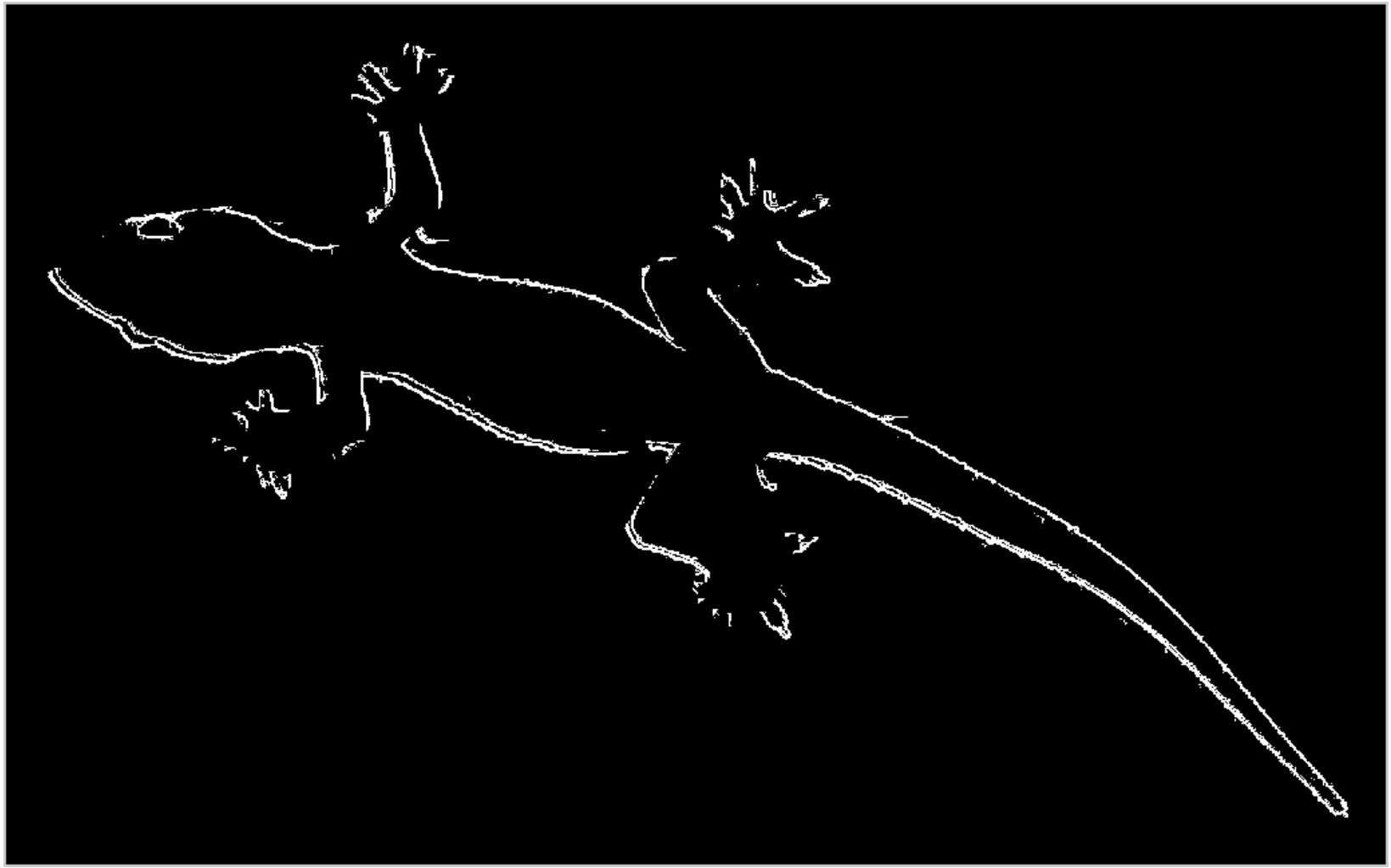
Results



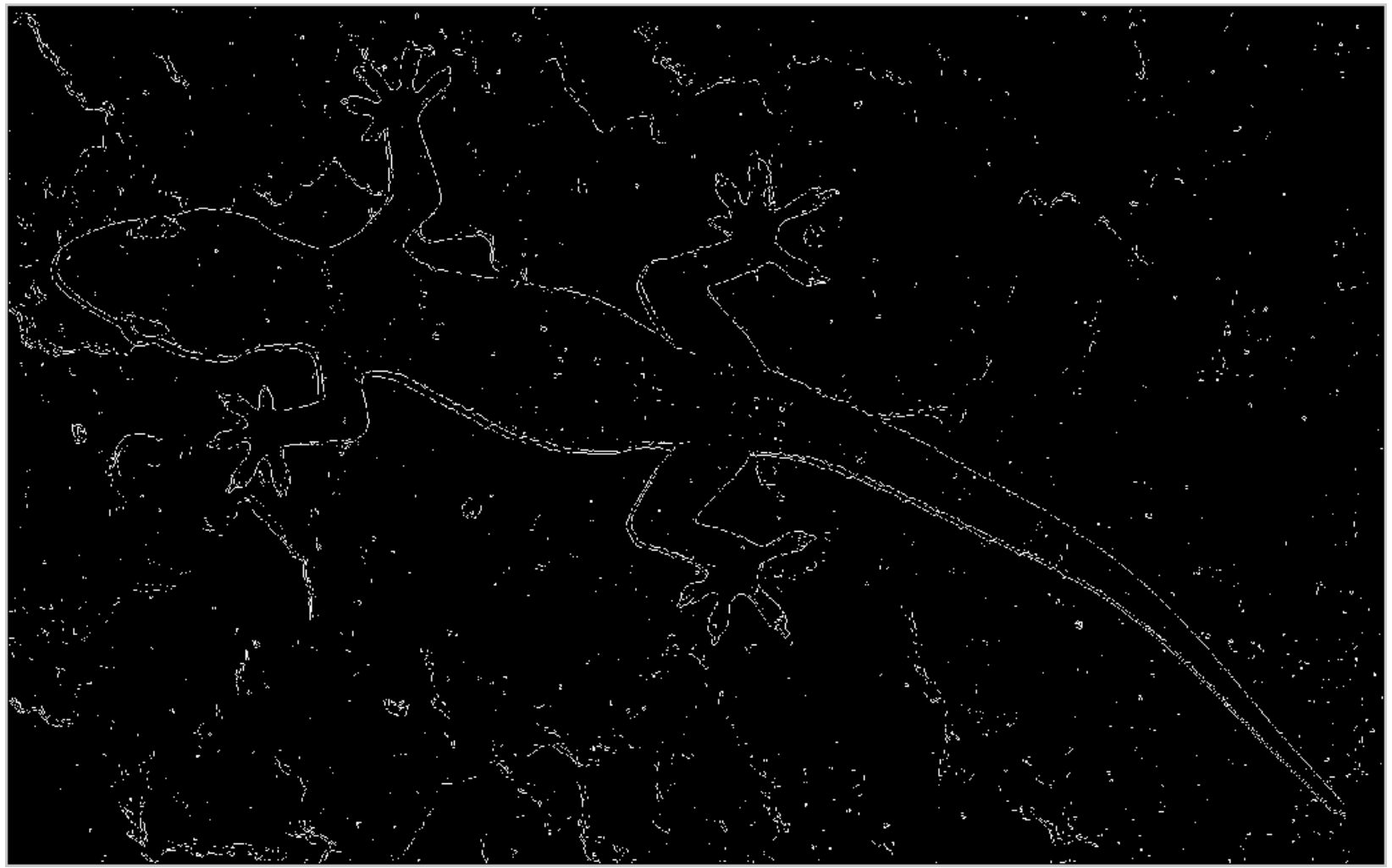
Results



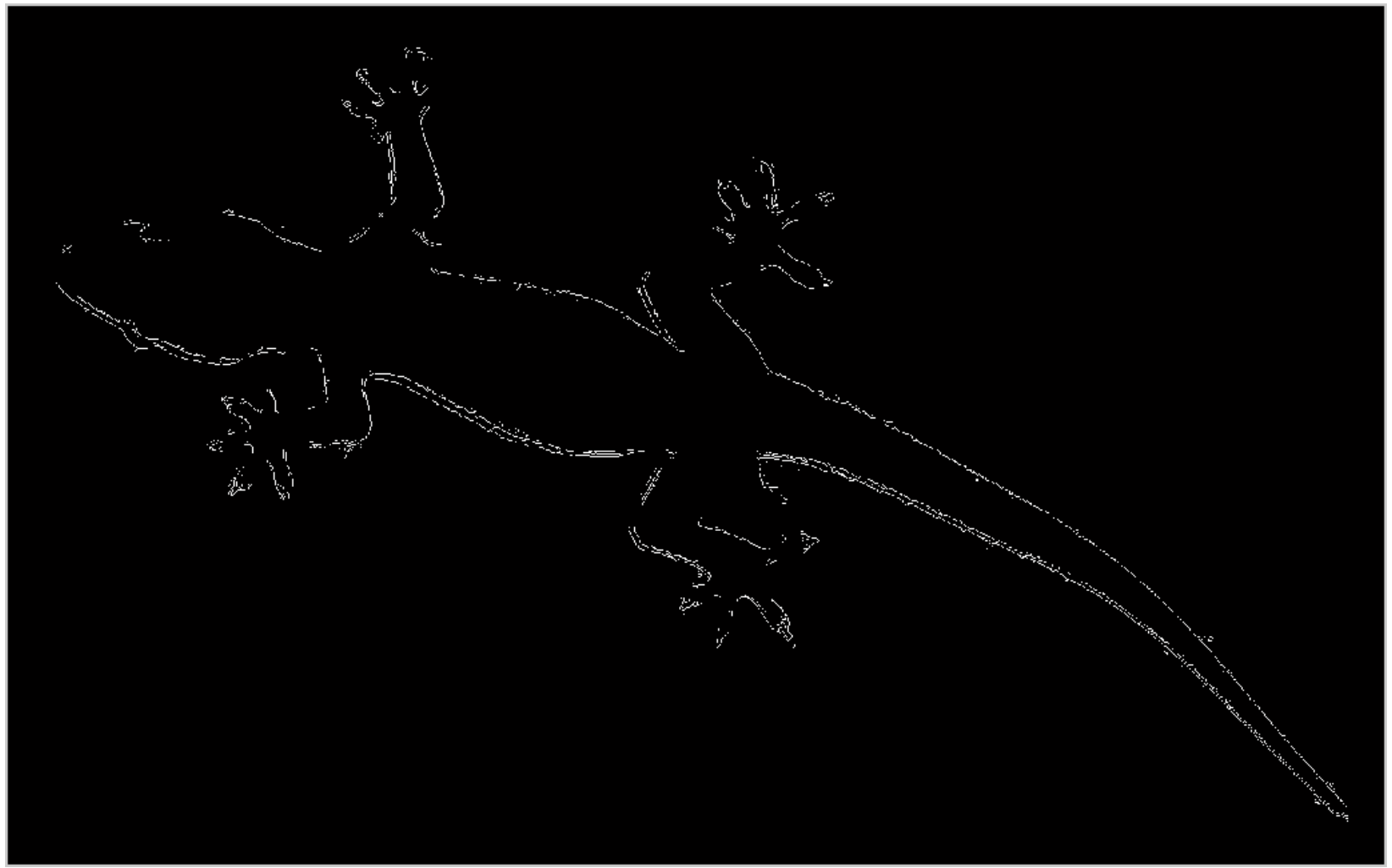
Results



Results - Bad



Results - Bad



Conclusions

Both methods require as input both curve data from the user and an edge map.

The proximity-based method demands (relative) precision from the former, in return for lesser dependency on the latter, while the intersection-based method has the opposite dependencies.

Conclusions

Both provide effective tool to filter out unneeded edges. Whether caused by noise or unwanted objects. Furthermore because the results are highly independent, both methods can be used together. For example we could use the proximity filter to increase the number of intersection points or limit the wild spread of the Hysteresis.

I believe the results shown a sufficient reason for additional research on interactive vision techniques.