Area Examination

Artificial Intelligence

May 8, 2006

Work each problem in a separate bluebook. Write the problem number and your PIN on the bluebooks. There are three parts (A, B, C), each with three problems. You are to complete six problems in total, two from each of the three parts. No reference materials.

1 Part A: Vision and Mathematical Foundations

Problem A:1: 3-D Range Data

Our lab's Minolta Vivid 900 features a high-resolution laser stripe ranger and a color camera so that superimposed color and depth "images" can be produced.

A: In as much detail as possible, explain the idea and geometry behind the light-striping range-finding technique (hint: it has nothing to do with the time of flight of a light beam).

B: You want to find "object boundaries" in a scene, so discuss how you would compute relevant edge features from the depth and color images. Unfortunately, the depth image is really a dense cloud of 3-D points, probably not in a nice "scan-line" order nor spaced evenly on a nice rectangular (x,y) grid.

C: Now how can Markov Random Fields be used to clean up and link together your noisy edge data? How can MRFs be used to put together the information from depth and color to improve the process of segmenting the scene into objects?

Problem A:1: Answer

A: Light striping can be thought of is a form of stereo in which the matching problem is simplified by limiting (to a unique pixel) the usual one-dimensional search for matching pixels. Another way to think of it is to note that the stripe on the scene is a plane in space. Now consider a line of sight from the camera center through an image location (x, y) that is on the image of the light stripe. That line intersects the plane in one point in space, and in fact that point in space is where the stripe falls on some surface in the scene. Calibrating the device thus amounts to knowing the light plane equation $Ax + By + Cz + D = 0$ and computing the homography between the image plane and the light plane (i.e. where the line of sight intersects the light plane in 3-D). Once that is done, hardware can detect the luminance edge caused by the stripe for every scan line in the image and compute the depth of all striped points in a video frame time.

B: Here insert all you know about color spaces, edge finders, edge linkers, and possibly raise the issue of what it might mean to be a "color edge", i.e. an edge defined in a 3-D vector space. Only hard part of this phase of answer is not running out of bluebooks and time. The rather interesting aspect is that pesky cloud of points, which from part A we see will correspond not to any 2-D grid but will look like the 3-D counterparts of the light stripes... 3-D points strung at intervals along locally approximately 'parallel' 3-D curves. One idea is to use the (X,Y) coordinates to texture-map the Z (depth) data onto a plane: getting Z values on a grid involves interpolating such a projection – like texture mapping. Another idea is to use commercial software to mesh, or triangulate, the 3-D data into a polygonal

surface and then use rendering tools to calculate the depth to the surface at X,Y grid points. After that, apply first phase of this answer.

C: You have to assume there are edgel configurations you like (such as two collinear edgels butting up against one another like — —), those you don't like (e.g., two noncollinear parallell edgels like =), and those that are unlikely (e.g. four edgels aiming at the same point (like +). Create an MRF with edgels as nodes, and labels of "belongs" and "does not belong". Let cliques consist of a neighborhood region around each pixel, assign clique potentials consistent with your prejudices as above, and relax the MRF to minimize the energy and maximize the probability of correctly labeling each edgel as to whether it should survive or not in the final answer. The idea of "coupled" MRFs is to have two such MRFs, one for the depth and one for color edges (which might have different rules for edgel-linking) communicate with one another, effectively adding another dimension to the clique space. Presumably you'd like the new clique potentials to encode your prejudice that a depth edgel can reinforce a color edgel of similar orientation at the same location and vice-versa.

Problem A:2: Stereo

Two cameras in the Z=0 plane, 10cm apart horizontally (in the X direction), both look out parallel to the Z axis and both agree on their vertical Y axis. 100cm away is a "picket fence" of narrow identical vertical stripes spaced every 2cm, which is all the cameras see.

A: What is the farthest and next-farthest distance (non-negative Z value) a stereo system will attribute to the fence? What is the farthest distance that has a disparity of different sign?

B: Consider adding a third stereo camera above the existing horizontal ones, gaze direction parallel too the other two. Can such trinocular stereo help resolve the depth ambiguity? Does it make a difference where the third camera is located horizontally with respect to the existing two?

C: Define the epipolar line (a picture is nice), say why it is useful in a stereo algorithm, and discuss some of the problems of matching along such a line and the applicability of dynamic programming to them.

Problem A:2: Answer A: The farthest the fence can be is infinity, if matching pickets have zero in the image disparity. Going right to 3-D and defining d as the "real world disparity", i.e. the distance along the fence between pickets matched by the algorithm, we get false depths by similar triangles: Z:10::Z-100:d (this gives the infinity result for d=10). So at the next smallest real-world disparity $d = 8$ we get $Z = 500$. When the disparity goes to its first negative value of -2 we get $Z = 500/6$.

B: Putting 3rd camera directly over either of the other two is useless. Anywhere else gives us three separate horizontal baselines, which will help disamiguation. Limiting the location to being between the orignal two cameras in X, half-way between them is the next-worse place, and any rational choice for the ratio of baselines is worse than an irrational choice, which is guaranteed to eliminate all ambiguity.

C: Given two cameras with camera centers O_1 and O_2 and a 3-D scene point P, these three points must lie in a plane Π. That plane intersects each image plane in an *epipolar line* that includes the image of the point P and the image of the other camera center. It is useful because any 3-D point on the line of sight from O_1 to P lies in Π and is thus projected onto the epipolar line in camera O_2's image. Thus the image of any bit of surface along that line of sight seen from O_1 has a corresponding image along O_2's epipolar line. The stereo matching problem thus collapses into a 1-D matching problem of matching points along the two epipolar lines, rather than, say, a 2-D patch-matching problem. Ordinarily we like to think of the cameras being carefully vertically aligned so the epipolar lines are just scan lines, and if we're careful we can get matching scan lines in each image to be at the same image height, so the matching works "as you think it should". But it's thanks to Old Mr. Epipolar Line that this simple-minded and intuitive approach works.

Now as we know the geometry is the easy bit and the matching is the hard bit in stereo. What you expect along your epipolar line is intervals where there are more-or-less corresponding pixels (but they may have been stretched out in one view), and also intervals of no match at all (where one camera sees something that is obscured by a foreground object from the other camera). We're looking for a maximal match, something like string matching, in which the dependence of the goodness of the match only depends on the goodness of the match of sub-lines to the Left and Right — there is minimal connectivity between the sub-problems. This (strongly) suggests dynamic programming because you pay DP to solve lots of subproblems, keep the sub-results around, and hook them up in the optimal way. More details and a nice picture quite welcome at this point.

Problem A:3: Finding Rectangles

You work for a company that uses aerial images to generate various sorts of maps. The company wants to identify buildings in such images. Since many buildings have rectangular components, the company wants you to write a program that can find rectangular regions in an image on the basis of edges. You do not know the orientation of any such regions i.e. they are not aligned with the image axes, and there may be buildings of many different orientations in the image. You also have only broad constraints (e.g. order of magnitude) on size and dimensions.

Assume you already have done a first stage of processing that, through a combination of edgel detection, linking, and analysis, has identified a set of line segments in an image. You want to find sets of line segments that are consistent with rectangular hypotheses. Unfortunately, although the rectangles of interest have line segments along the majority of their boundary points, the line segments found are not whole rectangle sides. Sides are often broken into a few segments. Nor do the segments always end at the corners. There is often premature termination, and sometimes overshoot or merging with another edge. You do, however, typically have good orientation, and accurate position on the axis perpendicular to the segment.

One approach is to try to identify potential corners on the basis of segments that approach each other at a 90 degree angle - even if they do not actually have corresponding endpoints. Consistent sets of such corners could then be used to hypothesize rectangles, that could be passed to a verification process.

A: Describe how you could locate such potential corners. How efficient can you make your process? Consider that a typical image may have several thousand line segments identified in it.

B: How many such corners does it take to hypothesize a rectangle? How costly is brute-force search over subsets of the requisite size? Describe a technique for making the search (much) more efficient.

C: How would you go about attempting to verify hypothesized rectangles?

Problem A:3: Answer: Finding Rectangles

One approach. There are certainly others that could work.

1. Organize the line segments by orientation. For each line segment, look at those that are near 90 degrees to it. Locate the intersection point. If it is close enough to each line segment using some measure that probably incorporates the length of the segment, hypothesize a corner at that location involving the specified segments. If a maximum scale is known, the image could aslo be organized by zones, with only segments in the same or ajoining zones being considered.

2. Because the ends of the segments do not necessarily correspond to corners, it takes 3 corners to hypothesize a rectangle. A brute force search over all triples of corners would be very costly. (n^3 in the number of corners) The process can be considerably improved by organizing the corners by orientation (say by the angle of the bisector). Then for a given corner, search for the clockwise adjacent one by considering only corners whose orientation is 90 degrees clockwise. Filter on plausibility (i.e. need the

associated segments to cover some minimum fraction of the side between the corners. For any such candidates, search for a third corner from candidates with the correct orientation. As in part 1, if there are known size constraints, the corners can also be organized into zones.

3. Verification could take the form of checking how much of the hypothesized rectangle's border is actually present in terms of detect line segments contained between the corners. This could either be done by locating whole segments that are close to the boundary using zone and orientation indexing, or by using an orientation sensitive chamfer match on segments projected back into the image plane. We might look for the 4th corner, but we also might want to find rectangles where the 4th corner is not detected for some reason. One could also drag in blob information and gradient direction at this point in an effort to ensure that the passed rectangles are associated with a single object, and not just hallucinated from miscellaneous lines.

2 Part B: Symbolic AI

Problem B:1 Machine Translation

Statistical machine translation system use Bayes rule to decompose a possible translation's probability into a product of translation model and language model probabilities:

$$P(\mathbf{e} \mid \mathbf{f}) = \frac{P(\mathbf{f} \mid \mathbf{e})P(\mathbf{e})}{P(\mathbf{f})}$$

Translation of a new French sentence $\mathbf{f}$ into English is performed by search for the English sentence $\mathbf{e}^*$ that maximizes this formula.

$$\mathbf{e}^* = \operatorname*{argmax}_{\mathbf{e}} P(\mathbf{e} \mid \mathbf{f})$$

A: How would you model $P(\mathbf{f} \mid \mathbf{e})$?

Answer: Independent lexical translation and uniform reoredering (IBM Model 1), phrase pairs, syntax trees; the possibilities boggle the mind.

B: How would you model $P(\mathbf{e})$?

Answer: n-grams

C: How would you model $P(\mathbf{f})$?

Answer: Trick question: you wouldn't, the denominator is constant with respect to argmax.

D: If you have a way of computing $P(\mathbf{f} \mid \mathbf{e})$, why not just use the same technique to compute $P(\mathbf{e} \mid \mathbf{f})$ directly, without using Bayes rule?

Answer: None of our translation models are very good, and ngrams focus probability mass on fluent English sentences.

Problem B:2: Deductive Question Answering Given:

1. Every number is divisible by itself.

2. If a number is not divisible by any number smaller than itself other than 1, then it is prime.
3. If a number is divisible by 2 then it is even.
4. The only number smaller than 2 is 1.

Question: *What prime number is even?*

A: Express the premises and an appropriate claim corresponding to the question in FOL. Assume that the domain of discourse is the natural numbers {1, 2, ...}, so you need not use an explicit "number" predicate. Restrict yourself to the predicates D(x,y) ("x is divisible by y"), E(x) ("x is even"), P(x) ("x is prime"), and S(x,y) ("x is smaller than y"), and use the constants "1" and "2" (with the conventional numeric meaning).

B: Then make the appropriate conversions and

C: produce a resolution/paramodulation proof of the claim, and extract the desired answer from the proof in the manner of C. Green (indicating the extraction with an *Ans*-literal).

Hints: You should obtain 3 clauses for statement (2). The proof can be done in 7 resolution/paramodulation steps, but there's also (unexpectedly!) a 6-step proof without paramodulation. If you find a proof significantly shorter than this, it is apt to be incorrect – recheck your premises and steps.

D: Is your proof a *set of support* proof? Why or why not?
Is it a *unit preference* proof? Why or why not?
Is it an *ordered* resolution proof? Why or why not?

Problem B:2 Answers

A: 1. $\forall x\, D(x,x)$
2. $\forall x\, [\exists y\, D(x,y) \wedge S(y,x) \wedge \neg\, y{=}1] \vee P(x)$
3. $\forall x\, [D(x,2) \Rightarrow E(x)]$
4. $S(1,2), \quad \forall x\, [S(x,2) \Rightarrow x{=}1]$

Question: $\exists x\ P(x) \wedge E(x)$

B: Denial of conclusion: $\forall x\ \neg P(x) \vee \neg E(x)$

C1. $D(x,x)$ from 1.
C2. $D(x,f(x)) \vee P(x)$ from 2.
C3. $S(f(x),x) \vee P(x)$ from 2. C4. $\neg\, f(x){=}1 \vee P(x)$ from 2.
C5. $\neg D(x,2) \vee E(x)$ from 3.
C6. $S(1,2)$ from 4 (could be omitted)
C7. $\neg S(x,2) \vee x{=}1$ from 4.
C8. $\neg P(x) \vee \neg E(x) \vee \underline{Ans(x)}$ from denial.

C: (Argument: From premises 1 & 3, 2 is even. From premise 4, only 1 is smaller than 2, so by premise 2 it is prime.)

C9. $E(2)$ r[C1,C5a]
C10. $P(2) \vee f(2){=}1$ r[C3a,C7a]
C11. $P(2) \vee P(2)$ r[C4a,C10b]
C12. $P(2)$ f[C11a,b]

C13. $\neg$E(2) $\vee$ Ans(2) r[C12,C8a]
C14. $\square \vee$ Ans(2) r[C9,C13a]

Thus, the answer is that 2 is an even prime.

D: The proof is: – *not* a set of support proof, as C8 is not used initially;
– *not* a unit preference proof, as C10 is non-unit, but e.g., C9 could have been used with C8;
– *not* ordered, as resolved literals are not leftmost in C10.

Problem B:3: Expectation Maximization

This question asks you to derive a general maximization step for EM in a setting where:

- Each item in your data has some observed and some hidden variables.
- All variables are discrete.
- We model each item as a belief network, with arbitrary structure, and arbitrary arrangement of hidden and discrete variables.

In the maximization step of the EM algorithm, we wish to chose parameters Θ that maximize the expected log probability of the hidden data Y and observed data X:

$$\begin{aligned} Q(\Theta; \Theta^{i-1}) &= E[\log P(X, Y \mid \Theta) \mid X, \Theta^{i-1}] \\ &= \sum_Y P(Y \mid X, \Theta^{i-1}) \log P(X, Y \mid \Theta) \end{aligned}$$

In our setting, the total probability of the data can be written as a porduct over vectors from our training set (indexed by n) and variables in the graph (indexed by i and j):

$$P(X, Y \mid \Theta) = \prod_n \prod_i P(X_i^{(n)} \mid \text{Par}(X_i)^{(n)}, \Theta) \prod_j P(Y_j^{(n)} \mid \text{Par}(Y_j)^{(n)}, \Theta)$$

where the probability for each variable is a multinomial conditioned on its parents with parameters θ:

$$= \prod_n \prod_i \theta_{i, X_i^{(n)}, \text{Par}(X_i)^{(n)}} \prod_j \theta_{j, Y_j^{(n)}, \text{Par}(Y_j)^{(n)}}$$

A: Assuming that probabilities for all hidden variables Y_j:

$$P(Y_j^{(n)} = \ell \mid X^{(n)}, \Theta^{i-1})$$

have already been computed for each data items n (and without worrying about how it was done), derive the sets of *expected counts* that must be accumulated by writing Q as a sum over variables i in the graph.

Answer:

$$\sum_Y P(Y \mid X, \Theta^{i-1}) \log P(X, Y \mid \Theta) = \sum_Y P(Y \mid X, \Theta^{i-1}) \log P(Z \mid \Theta)$$

$$= \sum_Y P(Y \mid X, \Theta^{i-1}) \left(\sum_n \sum_i \log \theta_{i, Z_i^{(n)}, \text{Par}(Z_i)^{(n)}} \right)$$

$$= \left(\sum_n \sum_i \sum_{Z_i^{(n)}, \mathrm{Par}(Z_i)^{(n)}} \log \theta_{i, Z_i^{(n)}, \mathrm{Par}(Z_i)^{(n)}} P(Z_i^{(n)}, \mathrm{Par}(Z_i)^{(n)} \mid X^{(n)}, \Theta^{i-1}) \right)$$

$$= \left(\sum_i \sum_{Z_i, \mathrm{Par}(Z_i)} \log \theta_{i, Z_i, \mathrm{Par}(Z_i)} \sum_n P(Z_i^{(n)}, \mathrm{Par}(Z_i)^{(n)} \mid X^{(n)}, \Theta^{i-1}) \right)$$

B: Use Lagrange multipliers to set each θ so as to maximize Q.

Answer:

$$\forall i \sum_{z_i} \theta_{i, Z_i, \mathrm{Par}(Z_i)} = 1$$

$$\frac{\partial}{\partial \theta_{i, Z_i, \mathrm{Par}(Z_i)}} \left(\sum_i \sum_{Z_i, \mathrm{Par}(Z_i)} \log \theta_{i, Z_i, \mathrm{Par}(Z_i)} \sum_n P(Z_i^{(n)}, \mathrm{Par}(Z_i)^{(n)} \mid X^{(n)}, \Theta^{i-1}) + \lambda_i \sum_{z_i} \theta_{i, Z_i, \mathrm{Par}(Z_i)} \right) = 0$$

$$\frac{\partial}{\partial \theta_{i, Z_i, \mathrm{Par}(Z_i)}} \left(\log \theta_{i, Z_i, \mathrm{Par}(Z_i)} \sum_n P(Z_i^{(n)}, \mathrm{Par}(Z_i)^{(n)} \mid X^{(n)}, \Theta^{i-1}) + \lambda_i \sum_{z_i} \theta_{i, Z_i, \mathrm{Par}(Z_i)} \right) = 0$$

$$\left(\frac{1}{\theta_{i, Z_i, \mathrm{Par}(Z_i)}} \sum_n P(Z_i^{(n)},\ mboxPar(Z_i)^{(n)} \mid X^{(n)}, \Theta^{i-1}) + \lambda_i \right) = 0$$

$$\theta_{i, Z_i, \mathrm{Par}(Z_i)} = \frac{\sum_n P(Z_i^{(n)}, \mathrm{Par}(Z_i)^{(n)} \mid X^{(n)}, \Theta^{i-1})}{\lambda_i}$$

3 Part C: Miscellaneous

Problem C:1 Closed Worlds

A: Let $\Delta_1 = \{\forall x.\, P(x) \Rightarrow (Q(x) \vee R(x)),\ P(A)\}$. Is CWA[$\Delta_1$] consistent? Justify your answer.

B: Let $\Delta_2 = \Delta_1 \cup \{\forall x.\, R(x) \Rightarrow \neg Q(x)\}$. Is CWA[$\Delta_2$] consistent? Justify your answer.

C: Find COMP[Δ_1;P] (see item A); COMP[Δ_1;Q]; COMP[Δ_2;P,Q] (see item B).

D: Does COMP[Δ_2;P,Q,R] exist? If so, find it, if not, explain why not.

E: With $\Delta = \{\forall x.\, (Q(x,A) \wedge P(x,A)) \Rightarrow Q(x,B),\ \forall x.\, (Q(x,A) \vee \neg P(x,A),\ \forall x. \forall y.\, (Q(x,y) \vee \neg Q(x,y)\}$, give reasons why COMP[$\Delta$;Q] is undefined.

F: Give a Δ' equivalent to Δ in item E (in the sense that every formula of Δ' is provable from Δ, and vice versa), where COMP[Δ';Q] exists. Briefly justify the equivalence, and find COMP[Δ';Q]. (*Hint*: Think in terms of clause form.)

Problem C:1 Answers

A: CWA[Δ_1] is inconsistent because it entails $Q(A) \vee R(A)$, but fails to entail either of Q(A), R(A), so that it contains $\neg Q(A)$ and $\neg R(A)$, inconsistently with $Q(A) \vee R(A)$.

B: CWA[Δ_2] is consistent because from $\forall$x.R(x) $\Rightarrow \neg$Q(x) and Q(A) $\vee$ R(A), we now have R(A). No other disjunction of ground atoms is entailed, because the only formula that could yield such a disjunction is $\forall$x. P(x) $\Rightarrow$ Q(x) $\vee$ R(x), but P(x) is not entailed by Δ_2 for any substitution for x other than A.

C: COMP[Δ_1;P] = $\Delta_1 \cup \{\forall$x. P(x) $\Rightarrow$ x=A$\}$

COMP[Δ_1;Q] = $\Delta_1 \cup \{\forall$x. Q(x) $\Rightarrow (\neg$R(x) An P(x))$\}$

COMP[Δ_2;P,Q] = $\Delta_2 \cup \{\forall$x. P(x) $\Rightarrow$ x=A, $\forall$x. Q(x) $\Rightarrow (\neg$R(x) $\wedge$ P(x))$\}$

D: COMP[Δ_2;P,Q,R] does not exist because Δ_2 is not *ordered* in P, Q, R. This can be seen from the axiom $\forall$x. P (x) $\Rightarrow$ Q(x) $\vee$ R(x), which makes it impossible to write separate formulas

$\forall$x. $E_1 \Rightarrow$ Q(x), $\quad \forall$x. $E_2 \Rightarrow$ R(x)

such that the first has no negative occurrence of R in E_1, or the second has no negative occurrence of Q in E_2.

E: COMP[Δ;Q] is undefined because it contains formulas that are not solitary in Q (the 1st and the 3rd).

F: The first formula in Δ can be seen to be a weakened form of the second (easily seen by writing the first in clause form). The third formula is a tautology. Thus these formulas are redundant , and Δ is logically equivalent to $\Delta' = \{\neg$Q(x,A) $\vee \neg$P(x,A)$\}$. Since there is no positive occurrence of Q in Δ', the completion w.r.t. Q adds the formula $\forall$x $\neg$Q(x,y), i.e., COMP[Δ;Q] = $\{\neg$Q(x,A) $\vee \neg$P(x,A), $\forall$x $\neg$Q(x,y)$\}$. (And now the first formula can be seen to be redundant, i.e., it is subsumed by the second.)

Problem C:2: Paramodulation

A: Find all paramodulants of the 2 clauses

f(g(x)) = h(A,x) $\vee$ P(A,B),
$\neg$(g(A) = f(g(B))) $\vee$ Q(h(A,B),f(B)).

(Incorrect paramodulants will lose marks, even if the correct ones are given as well, so be careful...)

B: Express in FOL: "If Twain is identical with Clemens, then Twain's father is identical with Clemens' father".

Put the denial of this statement in clause form, and derive a contradiction using resolution and paramodulation. (Use T for Twain, C for Clemens, and the term f(x) for the father of x.)

C: Formally, i.e., in terms of models $\mathcal{M} = (\mathcal{D},\mathcal{I})$, prove that the following special case of paramodulation is *sound* (you need not make reference to variable assignments):

$$\frac{\text{A = B, P(A)}}{\text{P(B)}}.$$

Since you are proving a semantic property of a syntactic operation on formulas, do not use any syntactic operations on formulas (like argument substitution), only semantic definitions!

Problem C:2 Answers

A: p[1a$_L$,2a]: P(A,B) $\vee$ g(A)=h(A,B) $\vee$ Q(h(A,B),f(B))
p[2a$_R$,1a]: g(A)=h(A,B) $\vee$ P(A,B) $\vee$ Q(h(A,B),f(B))
p[2a$_L$,1a]: f(f(g(B)))=h(A,B) $\vee$ P(A,B) $\vee$ Q(h(A,B),f(B))
p[1a$_R$,2b]: P(A,B) $\vee$ g(A)=f(g(B)) $\vee$ Q(f(g(B)),f(B))

B: $\neg$(T=C) $\vee$ f(T)=f(C)
Denial:
1. T=C
2. $\neg$f(T)=f(C)
3. $\neg$f(T)=f(T) p[1$_R$,2$_R$]
4. f(x)=f(x) permissible equality axiom
5. $\square$ r[3,4]

C: $\frac{\text{A=B, P(A)}}{\text{P(B)}}$ is soun d.

Proof. We need to show that all models of the premis es are models of the conclusion. Let $\mathcal{M}$ = ($\mathcal{D}$,$\mathcal{I}$) be a model of {A=B, P(A)}, i.e., $\models_{\mathcal{M}}$ A=B and $\models_{\mathcal{M}}$ P(A); i.e., A$_{\mathcal{I}}$= B$_{\mathcal{I}}$ and A$_{\mathcal{I}}\in$ P$_{\mathcal{I}}$. Hence B$_{\mathcal{I}}\in$ P$_{\mathcal{I}}$, and so $\models_{\mathcal{M}}$ P (B).$\square$

Problem C:3 Consciousness

Science is the art of measurement: What can be measured, how it can be measured, and what relationships exist between measureables.

If we want to discuss scientifically whether machines can be conscious or not, we need a way to quantify consciousness.

A: Propose a quantitative measure of consciousness for a system. It should be broad enough that it can be applied to both biological and machine systems (even if it assigns machines a value of 0). It need not be completely computational, if you think non-computational aspects are relevant.

B: How can the measure be evaluated for a specific system machine or living being. Your methods do not need to be immediately practical, but they should be applicable in theory, at the level of a thought experiment.

C: Using what you know about rocks, hurricanes, plants, dogss, humans, and existing robots, estimate where each lies on your consciousness scale. Fully justify your answers.