

The algorithm specified in the current MediaCapture draft is specified in terms of generating all possible configurations for a device, eliminating those that do not conform to the constraints, and picking the best one out of those that remain.

For many devices, which have a large number of configuration settings, this is obviously not an implementable strategy. This note gives some examples of more practical implementations.

Choosing among those strategies requires some knowledge about the properties to be constrained, and it's therefore more appropriate to give the algorithm in the more abstract terms.

Case 1: Low number of configurations

Consider a camera that can be opened in exactly two modes: 640x480@30 or 320x240@60.

In this case, implementing the algorithm exactly as described is appropriate. There are only two configurations to consider, and measuring the fitness of each is easy.

Case 2: Orthogonal configuration parameters

Consider a microphone that can be opened with echo cancellation turned on or off, automatic gain control turned on or off, and "level" set to any value between 0 and 1. These parameters are completely orthogonal, any combination is legal.

The "level" can't be easily represented by "all possible values", so instead we can represent it using a triple of "min", "max" and "ideal".

When evaluating constraints, "min" and "max" are moved whenever there are "min" or "max" constraints; if an "exact" constraint is encountered, "min" and "max" are set equal. When evaluating an "ideal" constraint, "ideal" is set to this value.

When choosing a final value, we have 3 cases:

- "ideal" is between min and max, inclusive: Choose "ideal"
- "ideal" is below min: Choose min
- "ideal" is above max: Choose max

One such triplet is used for each combination of the other parameters. This is a pattern that can be useful later, so let's name it a "min-max triplet".

Case 3: Linked configuration parameters

Alternative thoughts (cullen)....

For most case with width, height, aspect, newtons method would find the answer and be very fast. There might be a case with local optima that are not global but having a hard time constructing a case where newton's method does not work. If there are cases that don't work

for newtons, simulated annealing or any particle method would work. Roughly I would summarize that algorithm as pick 1000 random points inside the space. Pick the best one. Pick another 1000 points that are within 10% of the previous best. Next round drop the 100% by half and do a few more rounds where the amount of randomization reduces each time.

This is the harder case.

Consider a camera that (together with its drivers) can deliver any size video from 1x1 to 1024x768, in any aspect ratio.

This is 786.432 combinations, which then needs to be considered for each combination with other parameters - not very practical.

The parameters that can constrain resolution are height, width and aspect ratio (with the obvious linkages between them).

We can construct a min-max triplet for each of the constrainable values.

When resolving width, height and aspect ratio, the following algorithm makes sense:

- Constrain aspect ratio by width and height:
 - $\text{min-AR} = \max(\text{min-AR}, \text{max-width} / \text{min-height})$
 - $\text{max-AR} = \min(\text{max-AR}, \text{min-width} / \text{max-height})$
- Constrain width and height by aspect ratio and each other:
 - $\text{min-width} = \max(\text{min-width}, \text{min-height} * \text{max-AR})$
 - $\text{min-height} = \max(\text{min-height}, \text{max-width} * \text{min-AR})$
 -
- Repeat until no more changes.
- Pick an “ideal” for each of the 3 parameters, as for the single-parameter case above
- Compute the fitness distance for each combination of min, max and ideal (a maximum of 9 combinations).
- Pick the one with the minimum fitness distance of those combinations.

This will give a settings dictionary that satisfies all required constraints. For the case where all “ideal” parameters are inside the min/max range, it is also guaranteed to be the optimum case - fitness distance is zero.

For the case where one parameter had to deviate from ideal, it is guaranteed to produce a result that cannot be improved on by varying that single parameter (since the fitness function is monotonic on one side of “ideal”, and the parameter is up against a hard limit).

This value is permitted to be chosen by the wording in the spec (SHOULD choose the one with a minimum fitness distance, not MUST).

NOTE IN DRAFT: I *think* this is guaranteed to be true for all cases where 2 parameters deviate from “ideal” - but am pretty sure it's not possible to generalize it to the N-parameter case. And it doesn't say that it can't be improved by varying OTHER parameters.

Refinement: If a more optimal solution can be found, the direction of the optimal change can be computed by calculating the (multidimensional) derivative of the fitness function at the chosen point, making a new choice in that direction, and iterating. This is Newton's algorithm in multiple dimensions.

ANOTHER CASE

The case where the aspect is constrained to exact 4:3 and the the width and height can range can be handled with the algorithm above by treating the restriction as an exact required constraint to value 4:3 - that way, restrictions on width will be reflected into height and vice versa.