Is it possible to have a system where the gas costs of different EVM instructions is determined by some on-chain voting mechanism similar to the mechanism used to decide the block gas limit? Basically, for each block, the miner can adjust the cost of each instruction up or down by at most 1%.

It seems like such an approach will be able to avoid DoS vectors due to mispriced instructions, as well as accomodate future advancements in hardware that might make the relative costs of instructions drastically different (say, if elliptic-curve instructions in CPUs become commonplace, the elliptic curve instructions' gas cost could be downvoted).

Furthermore, it seems like such a mechanism can also be extended to contracts, or at least pure-function contracts that use a fixed amount of gas. Every time a miner runs such a contract, it could up/downvote a "discount factor" tied to the contract in the state tree that discounts the gas price of calling that contract from another contract. This would allow contracts whose real cost to the miner is less than the sum of its instructions (say, due to interpreter optimizations) to have a lower gas cost. It might also allow the elimination of precompiled contracts; we can instead implement then in EVM on-chain, but hard-code an optimized implementation and a discount factor in clients.

Is there something wrong with such a system? Would bad actors be able to manipulate the costs in a cryptoeconomically problematic way? I vaguely recall @vbuterin mentioning that he once considered dynamic contract costs but rejected it since it's not incentive-compatible, but I don't see any obvious reason dynamically pricing instructions and contracts can lead to abuse that doesn't also apply to dynamically adjusting the block gas limit.