We know The Single Anomalous Fact About The Runes that we have found is there is an Astronomically low number of double (Identical Twin) runes- i.e. an FF, or a BB, etc. For example see:  
  
[frequency count of all rune-pairs](http://imgur.com/eAVCeeC)  and  [probability of observed twin-runes](http://imgur.com/ZCytrje)  
  
Think about how small the estimate of ~10^(-69) is. How can it be explained? One way of thinking about this astonishing FACT is the runes have a “memory” of the proceeding rune. No-one has found any other significant correlations, i.e. no memory of the one before the previous one, or the one before that, etc. On all other tests I have seen the runes are sufficiently close to random (when taking all the remaining pages together).  
  
I think any sufficiently complex encryption method is always a possibility - but what we have with methods like Hill Cipher, simple streams, etc, is what I refer to as a "Fine Tuning Problem."  
  
(IMO) If your proposed method does not have a natural way allowing this \*memory\* (and not other types of \*memory\*) then you will have to play very hard to "fine tune" your algorithm to work (i.e. choose a very specific key). My suggestion is to pick algorithms that don't need fine tuned keys to give the Liber Primus rune distribution, but use algorithms with simple keys that can produce what we observe.  
  
**What methods could there be?**  
  
It's fairly \*\*easy\*\* to develop algorithms that can work - make your algorithm have a \*memory\* of the previously encoded rune. E.g. just add in a "recursive" part where you include the previous rune value in some way. Then try to make the condition for a repeated rune happen with ~1% frequency. A simple way to do that is with the runes that occur ~1% of the time in plaintext. [Here is a very simple recursive algorithm that can exactly generate the distributions we see:](http://pastebin.com/dZzEjm4j)  
 **There are others.**  
  
Doing this also gives you an inbuilt way to easily check your assumptions - just pick words from the cipher text that contain a "repeat" - there can now only be a few real words that have the assumed low frequency rune in the correct position.  
  
Or - turn it on it's head: Pick cipher text words with a repeat. Find all the dictionary words with a low frequency rune in the correct position  
  
(Low frequency runes are one like IO, AE, - you can easily work them out.) For your chosen word there will likely be only a few (if any) dictionary words. Then try and find ways to encrypt your chosen words into the CT.  
  
These are the types of methods I have been trying (when I get the time).  
  
Ultimately - I think it's still a pattern matching game, with all the same problems we've discovered for other methods, but this (at least imo) seems a more targeted approach for solving rune-pages.  
  
( Of course - fine tuning is possible - and it could be the answer - maybe the "deep-web-page" has a one-time-pad on it? ;-) )  
  
  
\*Comments, questions, suggestions, omissions etc ? please try [#cicadasolvers](http://webchat.freenode.net/?channels=cicadasolvers)  
  
MSGA