Resampling Algorithm Proposal

Mitchell Gilmore

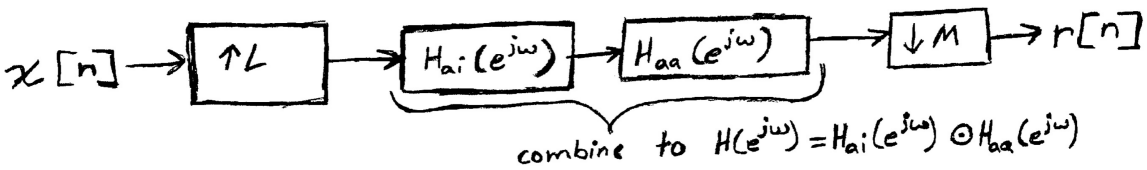
February 26, 2023

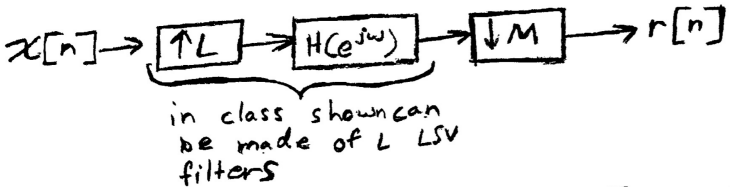
Abstract

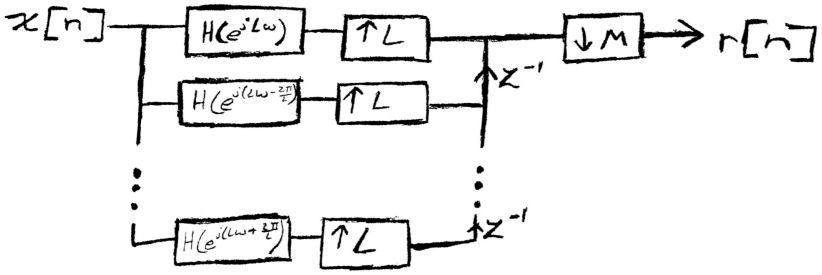
The purpose of this proposal is to suggest a resampling scheme that can be implemented to translate an arbitrarily sized finite signal from one sampling rate to another. Though the implementation is for a finite signal it will be written in such a way for easy translation for to use in a real time application.

This implementation will only implement a structure covered in class and will place the burden of an anti-aliasing filter design to the end user. For testing the filter for this application will be determined using prior software to determine an optimal filter. Future work may implement optimal filter selection, but it falls outside the scope of this proposal.

Diagram







Algorithms

Algorithm 1 Resampler Initialization

procedure RESAMPLER($f_{origin}, f_{target}, H_{aa}$)
    $L \leftarrow f_{target}$ ▷ Initializes the resampling factors L and M
    $M \leftarrow f_{origin}$
    while GCF$(L, M) > 1$ **do** ▷ loops through L and M until L/M is optimal
        $Z \leftarrow$ GCF(L, M) ▷ Determines The Greatest Common Factor
        $L \leftarrow L/Z$ ▷ Reduces L and M by a factor of Z
        $M \leftarrow M/Z$
    end while
    $self.L \leftarrow L$ ▷ Stores optimal L and M to object
    $self.M \leftarrow M$
    for i in range(L) **do** ▷ Partitions H_{aa} into L LSV filters
        $self.h[i] \leftarrow$ reverse$(H_{aa}[i :: M])$
    end for
    return $self$
end procedure

Algorithm 2 Resampler call

procedure RESAMPLER.CALL(x)
    $h_i \leftarrow 0$ ▷ Initializes filter index
    $o_i \leftarrow 0$ ▷ Initializes origin array index
    $t_i \leftarrow 0$ ▷ Initializes target array index
    $h_l \leftarrow$ len$(self.h[h_i])$ ▷ gets current LSV filter length
    while $o_i + h_l <$ len(x) **do** ▷ loops until out of data
        $t[t_i] \leftarrow \langle x[o_i : o_i + h_l], self.h[h_i] \rangle$ ▷ inner product between x and h
        $t_i \leftarrow t_i + 1$ ▷ sets data for next iteration
        $h_i \leftarrow (h_i + self.M) \mod self.L$
        $h_l \leftarrow$ len$(self.h[h_i])$
        $o_i \leftarrow o_i + \lfloor (h_i + self.M)/self.L \rfloor$
    end while
    return t
end procedure