Prolog for Artificial Intelligence

Consider the following problem to be solved using a Prolog program:

Given a closed planar polygon chain represented as a list of \( n \) vertices

\[
[v(x_1, y_1), v(x_2, y_2), \ldots, v(x_n, y_n)]
\]

compute the area of the enclosed polygon, and the orientation of the chain. The area is computed by the line integral \( \frac{1}{2} \int x \, dy - y \, dx \) where the integral is over the polygon chain. A naïve solution is given by the following program, which defines the predicate \texttt{area}. The goal \texttt{area(Chain, Area)} succeeds when \texttt{Chain} is the list of vertices, and the magnitude of \texttt{Area} is the area of the polygon bounded by the chain. The sign of \texttt{Area} is positive if the orientation of the polygon is anticlockwise and negative if it is clockwise:

\[
\begin{align*}
\texttt{area([X],0).} \\
\texttt{area([v(X1,Y1),v(X2,Y2)|VS],Area):-} \\
& \quad \texttt{area([v(X2,Y2)|VS],Temp),} \\
& \quad \texttt{Area is Temp + (X1 * Y2 - Y1 * X2) / 2.}
\end{align*}
\]

Explain how vertices are processed by this procedure. \[4 \text{ marks}\]

Why does this program execute inefficiently? \[3 \text{ marks}\]

Write an alternative definition that is tail-recursive and makes use of accumulator variables. \[10 \text{ marks}\]

Explain why your alternative definition executes more efficiently. \[3 \text{ marks}\]