🏛️ Arsitektur Aplikasi Utama

Struktur dasar aplikasi ini adalah kelas ImageProcessingApp yang mewarisi ctk.CTk (jendela utama dari CustomTkinter).

- **__init__(self)**: Ini adalah "otak" dari aplikasi.
    - **Modernisasi UI (Baru/Modifikasi):** Bagian ini telah banyak dimodifikasi. Anda mendefinisikan **palet warna** (self.COLOR_BG_DARK_1, self.COLOR_ACCENT_BLUE, dll.) dan **berbagai jenis font** (self.FONT_TITLE, self.FONT_BUTTON, dll.) secara terpusat. Ini adalah praktik yang sangat baik untuk konsistensi visual.
    - **Inisialisasi Variabel:** Menyimpan status gambar (original_image, processed_image), riwayat (history, redo_stack), dan referensi ke widget-widget penting (seperti label nilai slider).
    - **Tata Letak (Layout):** Menggunakan grid untuk membagi jendela menjadi dua bagian utama: control_frame (panel kontrol di kiri) dan image_display_frame (area gambar di kanan).
- **create_menubar(self)**: Membuat menu bar tradisional di bagian atas jendela (File, Edge Detection, Basic Ops, dll.). Bagian ini **juga dimodifikasi** untuk:
    - Menggunakan palet warna dan font yang baru agar terlihat konsisten dengan tema.
    - **Memperbaiki Menu Segmentasi:** Menu ini diatur ulang untuk menyertakan fungsi-fungsi baru yang ditambahkan.
- **Panel Kontrol (Kiri)**: Berisi tombol-tombol statis (Buka, Simpan, Undo, Redo, Reset) dan sebuah CTkScrollableFrame (dynamic_controls_frame). Frame ini adalah bagian penting, karena isinya akan **dihapus dan diganti** secara dinamis (clear_dynamic_controls) tergantung pada operasi apa yang dipilih pengguna dari menu.
- **Panel Gambar (Kanan)**: Menampilkan dua gambar: "Gambar Original" dan "Gambar Hasil Proses".

💖 Logika Inti: Pemrosesan & Riwayat

- **apply_filter(self, filter_func, *args)**: Ini adalah **fungsi terpenting** dalam logika aplikasi.
    1. Ia menyimpan gambar saat ini ke self.history (untuk Undo).
    2. Ia menjalankan fungsi pemrosesan yang sebenarnya (apapun yang dikirim sebagai filter_func, misal cv2.Canny atau cv2.blur).
    3. Ia memperbarui self.processed_image dengan gambar hasil filter.
    4. Ia menampilkan gambar baru dan memperbarui status tombol Undo/Redo.
- **Sistem Preview (Pratinjau)**: Untuk operasi dengan slider (seperti Brightness, Rotasi, Zoom), aplikasi ini menggunakan sistem pratinjau:
    1. Saat slider digerakkan, fungsi preview_... dipanggil.
    2. Fungsi ini menerapkan filter pada self.temp_image_for_preview (bukan pada gambar utama).

3. Hasilnya ditampilkan di panel kanan, namun **tidak disimpan** ke riwayat (history).
4. Hanya ketika pengguna menekan tombol "Terapkan", fungsi commit_... akan dipanggil, yang kemudian menggunakan apply_filter untuk membuat perubahan permanen.

- **undo_action(self) & redo_action(self)**: Menggunakan deque (antrian dua sisi) untuk memindahkan status gambar antara self.history dan self.redo_stack, memungkinkan fungsionalitas undo/redo.

✨ Fitur Baru dan Modifikasi Utama

Inilah bagian-bagian yang ditandai "[BARU]" atau "[MODIFIKASI]" dalam kode Anda, yang merupakan peningkatan utama:

1. 🎨 Kontrol Kontur Citra (Paling Signifikan)

Ini adalah perombakan total dari fungsionalitas kontur sebelumnya.

- **Impor Baru:**
  - from tkinter import colorchooser: Untuk membuka dialog pemilih warna bawaan Windows/Mac/Linux.
  - import re: (Regular Expression) Untuk memvalidasi input teks hex.
- **Helper Functions Baru:**
  - _is_valid_hex: Menggunakan re.compile untuk memeriksa apakah sebuah string cocok dengan pola #RRGGBB.
  - _ask_contour_color: Membuka colorchooser.askcolor(), mengambil warna hex, dan memasukkannya ke dalam CTkEntry.
  - _update_color_preview_from_entry: Memperbarui kotak pratinjau warna secara *real-time* saat pengguna mengetik di CTkEntry, dan mengubah border menjadi merah jika format hex tidak valid.
- **setup_contour_controls(self) (Modifikasi):**
  - Tidak lagi hanya tombol. Sekarang membuat UI yang kompleks di panel dinamis.
  - Membuat CTkEntry (self.contour_color_entry) bagi pengguna untuk mengetik kode hex.
  - Membuat tombol "Pilih..." (self.contour_color_chooser_button) untuk memanggil _ask_contour_color.
  - Membuat CTkFrame (self.contour_color_preview) sebagai kotak pratinjau warna.
- **commit_contours(self) (Modifikasi):**
  1. Mengambil string hex dari self.contour_color_entry.
  2. Memvalidasinya menggunakan _is_valid_hex.

3. **Mengkonversi Hex ke BGR:** Ini penting. OpenCV tidak mengerti hex. Kode ini mengkonversi string (misal "#FF0000") menjadi nilai (R=255, G=0, B=0), lalu menyusunnya sebagai tuple **BGR** ((0, 0, 255)) yang dimengerti OpenCV.
4. Memanggil apply_filter dengan fungsi worker _apply_contours_func dan memberikan warna BGR sebagai argumen.

2. 💾 Fitur Baru di Menu Segmentasi

Menu "Segmentation" telah diperluas dengan fitur-fitur baru yang kuat:

- **Kompresi Citra:**
  - apply_compression_preview: Mensimulasikan kompresi **Lossy (JPEG)** dengan kualitas sangat rendah (kualitas 10) untuk menunjukkan artefak kompresi.
  - show_lossless_info: Hanya menampilkan messagebox yang menjelaskan bahwa kompresi **Lossless (PNG)** tidak mengubah tampilan gambar, hanya ukuran file saat disimpan.
- **Steganografi (Menyembunyikan Pesan):**
  - setup_steganography_encode: Meminta pengguna memasukkan pesan rahasia menggunakan CTkInputDialog.
  - _text_to_binary: Fungsi helper untuk mengubah teks (misal "Hi") menjadi bit biner ("0100100001101001").
  - _steganography_encode_func: Fungsi worker yang sangat cerdas.
    1. Menambahkan "::EOF::" (End Of File) sebagai penanda akhir pesan.
    2. Iterasi melalui setiap piksel (Height, Width) dan setiap channel (R, G, B).
    3. Mengubah **LSB (Least Significant Bit)** dari setiap nilai channel piksel agar sesuai dengan bit pesan satu per satu.
    4. Ini mengubah gambar secara minimal sehingga tidak terlihat oleh mata manusia, tetapi pesan tersembunyi di dalamnya.
  - apply_steganography_decode:
    1. Membaca LSB dari setiap piksel dan channel, satu per satu.
    2. Membangun kembali string biner.
    3. Setiap 8 bit, ia mengubahnya kembali menjadi karakter.
    4. Ia terus membaca sampai menemukan penanda "::EOF::", lalu menampilkan pesan yang ditemukan.
- **Watermark (Visible):**
  - apply_visible_watermark: Memungkinkan pengguna menambahkan logo atau gambar lain sebagai watermark.

    - Meminta gambar kedua (_get_second_image_for_watermark), yang dibaca dengan IMREAD_UNCHANGED untuk menjaga **transparansi (alpha channel)**.
    - Mengubah ukuran watermark agar pas.
    - Menggunakan **alpha blending** (berdasarkan channel alpha watermark) untuk menempelkan watermark secara mulus di sudut kanan bawah gambar utama.

🧠 Fungsi Pemrosesan Lainnya (Inti Asli)

Kode ini juga berisi perpustakaan lengkap fungsi pemrosesan gambar standar, yang sebagian besar tetap sama:

- **Edge Detection:** Sobel, Prewitt, Robert, Laplacian, LoG, Canny, Kirsch.
- **Basic Ops:**
    - Aritmatika (Add, Subtract, Multiply, Divide) & Boolean (AND, OR, XOR, NOT) yang memerlukan gambar kedua.
    - Geometrik (Translation, Rotation, Zoom, Flip, Crop) yang kini memiliki UI slider kustom.
    - Colouring (Grayscale, HSV, YUV, Binary, RGB, CMY, Pseudo).
- **Enhancement:**
    - Histogram Equalization.
    - Smoothing (Blur, Median).
    - Sharpening (Highpass, Highboost).
    - Correction (Gamma).
- **Frequency Domain:** Fungsi-fungsi kompleks (_apply_filter_in_frequency_domain, _create_distance_matrix) untuk menerapkan filter ILPF, BLPF, IHPF, dan BHPF dengan memodifikasi spektrum frekuensi gambar.
- **Noise:** Fungsi untuk menambahkan berbagai jenis noise (Gaussian, Rayleigh, Erlang, dll.) ke gambar.

📈 Kesimpulan

Kode ini adalah evolusi besar dari aplikasi pengolah gambar standar. Ini bukan lagi hanya kumpulan filter, tetapi sebuah **alat yang matang** dengan:

1. **UI yang Profesional:** Berkat palet warna dan font kustom yang terpusat.
2. **UX yang Lebih Baik:** Penggunaan pratinjau interaktif (slider) dan kontrol dinamis.
3. **Fungsionalitas Canggih:** Penambahan fitur segmentasi modern seperti **Steganografi LSB**, **Visible Watermark** (dengan alpha-blending), dan **Pemilih Warna Hex** yang divalidasi re untuk Kontur.