ChatGPT

AI Bot Reference: Saitama (The Strongest Battlegrounds)

Core Combat Mechanics

Stuns, Hit Reactions, and Ragdoll Evasives

In **The Strongest Battlegrounds** (TSB), any time a character takes damage they enter **hit stun** – a brief period during which they cannot move, attack, dash, or block [1]. This is the basis of comboing: as long as each hit lands before the previous stun ends, the opponent cannot respond. The duration of stun depends on the attack; for example, basic M1 punches induce short stun (~0.1–0.2s), whereas certain heavy moves or multi-hits lock the opponent in longer animations. **Consecutive Punches**, for instance, keeps the target stunned for its entire multi-punch duration [2].

If a character is knocked **ragdoll** (e.g. slammed to the ground), they normally stay down ~2 seconds, but players have one escape: a **Ragdoll Cancel** (informally "evasive"). By performing a **Side Dash** or **Back Dash** while down, a player can **instantly get up** from ragdoll [3]. This "evasive" dash lets them avoid true follow-ups. On standing up from a ragdoll, the player gains about **0.5 seconds of immunity to stun** (cannot be immediately re-comboed) [4]. However, there are **limits**: you cannot use a Forward Dash to get up (only side/back work) [5] [6], and if you were put in ragdoll by certain perfectly-timed attacks, you may not get the chance. Notably, a **"true downslam"** (an aerial downward slam timed immediately after an M1 hit) can deny the opponent any chance to dash-escape [7]. For example, Saitama can do 2 M1s, then jump and land the 3rd M1 as a downslam *before* the prior hit-stun ends – this **true combo** knocks the enemy down while they are still stunned, preventing any dash out [7]. Similarly, the **4th hit of an M1 chain** for most characters causes a knockdown; if it lands, it briefly locks out evasives (~0.2s) so the combo can be extended [8]. These mechanics mean a well-timed combo can be "inescapable" without using an ultimate, as long as the attacker times launches/slams during stun periods.

An AI bot should **track the enemy's evasive availability**. Since dashes have cooldowns (details below), if the opponent **recently used a side/back dash** to escape a combo, the bot has a window (until that dash cooldown refreshes) where the enemy cannot ragdoll-cancel again. Typically, side dashes refresh every ~2 seconds [9], so an enemy effectively gets at most one escape per combo if you re-engage quickly. The bot can also infer an enemy used an evasive if it sees them **instantly stand up** from a knockdown or suddenly flip out of a combo. At that moment, it should note that the opponent's "get-up dash" is on cooldown. If the enemy has **no evasive available**, the bot can attempt longer true combos (see **Combo Decision Logic** below). If the enemy *does* have an escape ready, the bot might favor shorter combos or baiting the evasive first.

Blocking and Punish Windows

Players can hold **F to block**, reducing move speed and guarding a 180° frontal arc [10]. Light attacks (like basic punches) and some skills are completely nullified by a block, but **certain moves guard-break**.

Saitama's **Normal Punch, Shove, and Uppercut** are all *unblockable*, meaning they will go through a guard and strike the opponent [10]. (Consecutive Punches, however, **can** be blocked, as can the shockwave portion of Normal Punch [11].) Even when blocking works, there are **block stun and recovery** timings. After blocking an attack, a defender has a small stun and the attacker has some recovery – some moves are *safe* (attacker recovers first) and others are *unsafe* (blocker can punish during the attacker's end-lag) [12]. For example, many heavy skills in the game leave the user vulnerable if blocked or whiffed [13]. With Saitama, **Consecutive Punches** will leave you wide open if the enemy blocks the barrage (since you are stuck in the punching animation), and **Normal Punch** has a long recovery if it misses. As a rule, **missing the final hit of an M1 chain is extremely unsafe** – if the 4th M1 misses or is blocked, **the attacker is stunned for a full second** [14], basically guaranteed punishment. An AI should therefore avoid throwing out the last M1 if it's likely to miss or be blocked; it's often smarter to stop at 3 M1s and then use a combo extender or reposition, rather than whiff a 4th swing and eat a counterattack.

Canceling: In TSB, you generally cannot cancel out of attack animations (no mid-move dash cancel, etc.) except by using specific character counters or being hit. Once you commit to a move, you must endure its end-lag before you can block or dash. There is a **short delay after any M1 during which you cannot block** – this prevents instant block-cancel of every punch [14]. This delay increases with later punches (the 3rd punch has more recovery than the 1st, etc.) [14]. Thus, combo timing is important: if the bot throws out a move, it should be aware it can't instantly bail out. Conversely, the bot can exploit enemy end-lag: if it sees an opponent whiff a big move or the last M1, that is a prime moment to dash in and punish (their stun/lag is a *true punish window* [15]). Also note that a successful **Perfect Block** (timing F just as a hit lands) stuns the attacker briefly and gives the defender a powered-up next punch (Critical Hit) [16] [17]. This is a high-level mechanic; an AI could listen for the distinct *"crack"* sound of a perfect block (critical hit cue) [18] and then be wary – the opponent's next M1 will deal triple damage. In general, the bot should prioritize **not being predictable** with its own M1 strings (to avoid giving opponents easy perfect blocks or counter windows), and capitalize when the opponent over-extends.

Movement Constraints and Animation Cues

Most attacks in Saitama's base kit **root the user in place** during their execution. While throwing a punch (M1 or skill), you cannot walk or reorient freely. For example, **Normal Punch's** long wind-up literally has Saitama standing with fist cocked for ~1 second [19], and **Uppercut** similarly has a noticeable pre-punch delay (~0.8s animation) [20]. These animations are visually distinct: *Normal Punch* has Saitama pull his right arm far back before a straight body blow, *Uppercut* has him lower and then swing upward, *Shove* has a quick elbow strike motion, etc. An AI can use these **animation cues** to identify actions – for instance, if an enemy Saitama plants his feet and winds up a big punch (and you see a shockwave gathering), that's the **Normal Punch** startup, so you'd better **dodge** because it's unblockable. Similarly, the flurry of rapid swings with wind pressure effect indicates **Consecutive Punches** is in progress (don't walk into it; maybe try to block then counter after). The bot controlling Saitama itself can also leverage known **timing patterns**: e.g. the M1 punches come out very fast (~0.25s per swing) [21], whereas a move like Shove finishes in ~0.3s [22] and Consecutive Punches runs about 2 seconds [19]. These durations (summarized in the script's data) help the AI scheduler not overlap inputs – it should wait until the current move's animation (and hit registration) is done before the next action.

Another constraint is the **M1 combo timer**: if you wait too long between consecutive punches, the chain "resets." In practice, Saitama's combo must be continued within ~0.6 seconds or the next click will start back at the first punch [23]. Using a skill like Consecutive Punches **pauses the reset timer** [24], which is why

skilled players weave it into combos (more on that below). The bot should be aware that after a gap (due to spacing or dodging), it might have to restart the M1 sequence.

Visual/audible effects are also key cues: when Saitama's **Normal Punch** lands, it creates a loud boom and shockwave; even a **near miss** triggers a shockwave hitbox (10.5% damage) that rolls opponents [25]. If the bot sees an enemy suddenly ragdoll and go rolling from an unseen hit, it might infer a shockwave caught them. **Uppercut's** hit launches the victim with a distinctive *whoosh* and sends them flying upward (clear visual of a soaring target). **Shove** produces a unique knockback where the target is sent tumbling (and a successful reflect of a projectile with Shove might be noticed via the reflected object flying back) [26]. The AI can also monitor the **UI indicators** for its own status: cooldown icons on the hotbar shrink/fill while recharging. For example, each ability slot has a radial or vertical fill bar – if the icon is dimmed or a bar is visible, that skill is still cooling down [27]. The bot can read those frames to know when its moves are ready (or use an internal timer since it knows the exact cooldown seconds). This ensures the AI only attempts a move when available. Additionally, a **block** is visibly telegraphed (character holds arms up defensively), so the bot can detect if an enemy is currently blocking and consider using an unblockable or maneuver around them (see Dash section).

In summary, Saitama's combat follows typical fighting game logic: **keep the enemy in stun**, watch for their escape attempts, use unblockables against turtles, and leverage the timing and animations of moves to your advantage. Next, we detail the movement and dash system which is critical for closing gaps and repositioning in combat.

Dash Behavior and Orientation Logic

Movement in TSB is augmented by a dash system (key **Q** on PC) that allows quick bursts in any direction to evade or initiate. Dashes have three variants: **Forward Dash**, **Back Dash**, and **Side Dash**, determined by input. For a human player, pressing W+Q yields a forward dash, S+Q a backdash, A or D + Q side dashes [28] [29]. An AI bot can achieve the same by adjusting the character's facing or movement vector before triggering Q. **All dash types share an important cooldown mechanic**: Forward and Back dashes use the same cooldown (dashing forward puts your backward dash on cooldown and vice versa) [30]. This cooldown is roughly **5 seconds** long in current TSB [31]. Side dashes have a much shorter cooldown, about **2 seconds** [9]. (In other words, you can quick-step side to side more frequently, but you can't spam the long forward lunge.) The distance covered also differs: forward/back dashes travel a **long distance** in a straight line, while side dashes are a shorter sidestep (and actually become even shorter if your health is low, to prevent runaway tactics) [32].

Forward Dash: Saitama's forward dash propels him toward where he's moving/facing, with a quick sprint animation (he even swings an arm during the dash). This dash can be used offensively – in fact, if you collide with an opponent during a forward dash, the game will perform an **automatic attack**: your character stops and does a dash-punch that deals a small amount of damage (~4%) [33]. Hitting a very nearby enemy with a forward dash can cause them to **roll on impact**, taking ~5.3% and being knocked away [34]. For Saitama, that can function as a combo starter (dashing into an opponent then following up while they're rolled). The forward dash has a travel time of roughly 0.7–0.8 seconds in Saitama's case [35] [36], making it a fairly quick gap-closer but not instant. One downside: **Forward dash cannot be used as a ragdoll escape** (the game disallows using the forward lunge while knocked down) [5], likely because it has offensive properties. Only side/back dashes will immediately get you up if you're downed.

A unique aspect the AI can exploit is **orientation control during forward dashes**. Skilled players sometimes perform a slight turn mid-dash to end up beside or behind the opponent; the bot can do this even more precisely. In fact, our AI design includes a **"close-orbit" mechanic**: if Saitama forward-dashes at an enemy and ends up extremely close (within ~2 studs), the bot will temporarily **lock orientation 90° to the target** for ~0.3 seconds [37], effectively **circling around** the opponent before re-facing them [38]. This can confuse the enemy and position Saitama to their side or back. After this brief orbiting strafe, the bot re-orients toward the target automatically. The result is that a point-blank forward dash doesn't overshoot past the enemy; instead Saitama will sort of pivot around them, maintaining close range. This is useful for setting up something like a **Normal Punch from behind** or just staying on the opponent who tried to sidestep. (It's worth noting that the game's own behavior causes a close-range forward dash to stop on hit and make the opponent roll, as mentioned – the orbit technique is our bot's enhancement to keep sight on the target during that moment.) [38] [37]

Back Dash: A back dash is executed by Saitama flipping backward twice, creating distance. Back dashing has some **invulnerability frames** in its very first moments – the player is immune to M1 hits and similar during the initial backflip frames [39]. This makes backdash a great defensive move to avoid a hit at the last second (even if the attack is already in motion, you might evade the damage). Back dash shares the 5s cooldown with forward dash [31], so using one means you can't use the other immediately. Importantly, **back dash *can* be used while ragdolled** to get up quickly [39]. Many players intentionally backdash as a get-up move because of the brief invulnerability – it not only stands you up, but also dodges any immediate follow-up attack from your opponent. The AI should utilize back dash in emergencies (e.g. to escape a combo if its own evasive is available) or to disengage when low on health. Offensively, backdash is less used, but some advanced maneuvers involve backdashing then immediately countering if the opponent whiffs an attack.

Side Dash: Side dashes are quick lateral steps (left or right relative to movement). They refresh every 2 seconds and cover a modest distance [9]. Despite shorter range, they are extremely versatile: you can use side dash to **dodge linear attacks**, to **reposition around a blocking enemy**, or as a small combo linker. In fact, **dashing to an enemy's side or behind** is a common tactic to bypass their block (since block only protects front 180°) – players practice techniques like *"C-turn"* or *"L-turn"* dashes to slip behind a guarding opponent and then punish [40]. Our bot can do this instantaneously: if it detects the enemy turtling (holding block), it can perform a quick side dash around and attack from behind where the target is exposed. The side dash is also crucial for **combo extension** after launching an opponent. For example, after Saitama uppercuts someone into the air, the bot can **side dash forward** to where the target will land, allowing it to catch them with an M1 the moment they hit the ground (before they can recover or dash away). This is referenced as a *"side-dash catch"* – essentially moving in and re-engaging as the enemy is coming out of a launched state [41]. Side dashing does have slightly variable efficacy based on health (lower HP = slightly shorter dash) [32], but the bot can account for that if needed (the difference is shown in a linear graph in the game wiki).

In implementation, the AI uses dashes both **defensively** (to evade big moves or escape combos) and **offensively** (to close distance or extend combos). It will maintain an internal timer for each dash direction. For instance, after using a forward dash, it sets a 5s cooldown on both forward/back in its blackboard state [42]. After a side dash, 2s cooldown for side. This prevents trying to dash again while it's unavailable. The bot also aligns the camera/character before dashing to ensure the dash goes in the intended direction (e.g. face directly away from target and press Q to simulate a backdash). Overall, **mobility is key** – experienced players prioritize movement over flashy combos [43], and a good Saitama bot should do the same. Constant

unpredictable dashing makes you a harder target and can set up your own attacks. The bot will use short side-dashes to dodge in and out of M1 range and forward dashes sparingly when it sees an opening or needs to chase a fleeing opponent. With dashes and core combat covered, we can now detail Saitama's **specific abilities** and how they fit into combos.

Saitama's Ability Details (The Strongest Hero Base Kit)

Saitama's base kit (The Strongest Hero) consists of a rapid **M1 punch combo** and four special moves bound to keys 1–4 by default: **Normal Punch**, **Consecutive Punches**, **Shove**, and **Uppercut**. He also has a contextual **Wall Combo (Light Sneeze)** that triggers under certain conditions. All moves and their properties are summarized below:

Move	Damage	Cooldown	Properties & Usage Notes
M1 Punches (4-hit chain)	~14% total (approx. 3% +3%+4% +4% per hit) [44]	*None* (basic attack)	Four fast punches alternating hands [45]. The **fourth punch knocks the opponent down** (hard ragdoll) if it lands. Each hit stuns briefly, allowing true follow-up by the next hit. **Warning:** The 4th hit has a longer wind-up/recovery; if it **misses or is blocked, the user is self-stunned ~1 second** [14] (highly punishable). To avoid that risk, players often do 2–3 M1s then use a special move to extend the combo. After a full 4-hit combo, the target will be knocked away, ending the combo (unless near a wall, see Wall Combo). Note: if you wait >0.6s between M1 hits, the chain resets to the first punch [23].

Move	Damage	Cooldown	Properties & Usage Notes
Normal Punch (“One Punch”)	**31%** (20% direct + ~11% collision bonus) [46] + *Shockwave:* 10.5% [25]	**20 s** cooldown [47]	Saitama’s signature straight punch. **Unblockable** – it will break guard and hit anyone in front of Saitama. Long **wind-up (≈1s)** [19] where Saitama draws back his fist (making it telegraphed). On hit, deals massive damage and blasts the opponent far across the map. If the target is near a wall or floor when struck, they take ~11% extra damage from the impact [48] (total ~31%). The punch also creates a **shockwave** that travels forward; this deals 10.5% and causes a rolling knockdown to anyone *not* directly hit (or if the main punch was avoided) [25]. *Can hit multiple enemies* in a line with the shockwave. **Can be countered:** If an enemy has a counter ability active, they can reflect or negate Normal Punch (so the bot should be careful using it against characters in counter stance) [49]. **Best used as a combo finisher** when the enemy is already stunned or ragdolled (true combo into Normal Punch ensures it lands). Because it’s unblockable and high-damage, often the bot will finish a sequence with NP to either KO or severely hurt the opponent. Avoid using raw from neutral unless you predict the opponent’s approach, due to the long startup.

Move	Damage	Cooldown	Properties & Usage Notes
Consecutive Punches (“Consecutive Normal Punches”)	**14.5%** (multi-hit total) [50]	**15 s** cooldown [51]	A **barrage of rapid punches**. When activated, Saitama unleashes a flurry of blows over ~2 seconds, during which the target is caught in hit-stun if the first hits connect [2] . This move effectively **locks the opponent in place** if it lands, acting as a combo extender. It also **halts the M1 combo timer** – your M1 chain won’t reset during the barrage [52] . In practice, you can do a couple M1s, use Consecutive Punches, then continue with M1s as the wiki notes (“it can be added in any combo” [53]). **Blockable:** if the opponent is blocking, they will negate this move (and you’ll be vulnerable after the punches finish) [54] . Thus it’s best used when the enemy isn’t blocking or has no time to react (e.g. mid-combo). All hits of CP together do ~14.5% damage, and it does not knock the opponent away – they remain in front of you, which is perfect for follow-ups. **Combos:** Due to a slightly shorter stun on the final hit of CP (compared to a normal M1), a common combo is to only do **2 M1s, then CP, then continue with M1s** [24] . The bot will use CP as a key **combo extender** whenever it’s off cooldown, typically in the middle of a string to rack up damage and prevent the target from escaping. (Rarely, a server bug allows an “M1 reset” during CP, fully refreshing your combo – but the bot cannot rely on this random occurrence [52] .)

Move	Damage	Cooldown	Properties & Usage Notes
Shove (“Normal Shove”)	**6.9%** (single-hit) [55]	**10 s** cooldown [55]	A swift **elbow strike** that **cannot be blocked** [56]. Shove is a utility move with lower damage but unique behavior: it **knocks back a standing opponent with a roll**. Enemies hit by Shove are sent tumbling backward (in a short **rolling ragdoll** state) [57]. If the target was already in a roll or mid-ragdoll (for example, you hit them with Shove *after* a downslam or while they’re getting up), Shove will instead **fully ragdoll** them on the spot [58] – basically slamming them down if they were rolling. This makes Shove a powerful **combo tool**: you can use it to catch an opponent who just got launched or is trying to regain footing, and force them into a ragdoll. On a normal standing hit, Shove causes a brief stun and a roll away; you can combo into other moves if you time it such that you reach them as they exit the roll. **Projectile reflect:** Shove, if timed right, will reflect throwable objects/projectiles back at the sender [26] (a fun bonus: e.g. reflecting Genos’ trash can projectile). **Can be countered** (it’s a physical hit). Shove has **very fast startup (~0.3s)** [22], so it’s great for interrupting enemies or quickly punishing a whiff. In combos, players often use Shove after a few M1s because it’s unblockable (ensuring the combo continues even if the opponent tries to hold F) and it puts the enemy into a roll that can be followed up. The bot will frequently do **M1→M1→Shove** as a string. After Shove connects, one tactic is to **hold the M1 button** – as soon as the enemy comes out of the roll, you’ll automatically punch, often hitting them the moment they stand (this is referred to as a “shove tech” leading into a **mini-uppercut** follow-up, described later) [59]. Overall, Shove is one of Saitama’s **combo extenders** (it’s noted he has 3 extenders in base kit [60]: Shove, Consecutive Punches, and Uppercut) and also doubles as a counter to blocking or projectile strategies.

Move	Damage	Cooldown	Properties & Usage Notes
Uppercut (“Normal Uppercut”)	**15%** (single-hit) [61]	**20 s** cooldown [61]	A powerful **upwards punch** that **launches the enemy high** into the air [62]. This move is **unblockable** (armor-breaking) [63]. On hit, it sends the opponent straight up and then they fall to the ground, ending in a rolling state on landing [62]. Uppercut has a noticeable **wind-up (~0.8s)** as Saitama pulls back and then uppercuts [20], so it can be dodged or interrupted if predicted. It **can hit grounded opponents** – if someone is lying on the floor (ragdolled), Uppercut will still pick them up off the ground and launch them [64]. This makes it useful if you down an opponent near you; you can immediately Uppercut to send them back up (*downslam → Uppercut is a common combo*). After being launched by Uppercut, an enemy can **use an evasive mid-air** *during the fall* to escape [65] – this is something to watch out for. If they have a side dash off cooldown, they might mid-air dash recover. For this reason, Uppercut is best used when you know the opponent’s evasive is unavailable, or as a true combo sequence where they have no time to react. The bot can also mitigate escape by **pursuing the launched enemy with a dash**: e.g. perform Uppercut, then **Forward Dash or Side Dash** immediately to where the enemy will fall, so that if they *don’t* evasive, you are in position to continue with M1 as they hit the ground (this is the “uppercut dash catch” technique) [66] [67]. Uppercut is Saitama’s third combo extender – it can continue a combo by bouncing the opponent into the air for further follow-ups when they land. However, using it mid-combo carries risk if the opponent still has a cancel; thus the bot often saves Uppercut for either finishing a combo (if enemy can’t escape) or after a downslam to extend. Like other big moves, it is counterable by counter-skills.

Move	Damage	Cooldown	Properties & Usage Notes
Wall Combo: Light Sneeze	**12%** (plus M1 follow-up possible) [68]	**~6 s** cooldown [69] (contextual)	The **Wall Combo** is a special move that triggers when you perform a **Forward Dash immediately after a full M1 combo** *and* the enemy's back is to a wall or solid object [70]. Every character has a unique wall finisher – Saitama's is an comedic **"Light Sneeze."** If you punch an opponent into a wall (i.e. you land your 4th M1 such that they are knocked into a wall behind them) and quickly press W+Q, Saitama will **shoulder-bump the enemy into the wall and then sneeze on them**, causing an explosion that deals 12% and knocks them down [69] [70]. This essentially "wombo-combos" off the wall collision. Light Sneeze has a short cooldown (can't be done again for several seconds) and slightly **leaves Saitama vulnerable at the end** ("mild end-lag") [71]. The move will send the opponent flying *backwards* if the wall isn't breakable (most map edges) or *through it* if it is destructible [72]. After a wall combo, the opponent ends up knocked down on the ground (just like a downslam) [73]. The game notes you **can follow up with an M1** after the sneeze if you time it perfectly as they get up [74]. Our bot will attempt this: after executing the wall combo, it will "hold M1" to try to land an immediate hit on the downed enemy as they stand – effectively continuing pressure. Wall combos are situational but very strong: Saitama's wall combo in particular is noted to be *inescapable and easy*, dealing ~59% total when combined with the M1 string leading into it [75]. The bot should prioritize wall finishes if the opportunity arises (enemy cornered against a wall), as it's free damage plus a knockdown.

Note: Saitama's kit also includes a powerful **Awakening (Serious Mode)** with new moves (e.g. Serious Punch, Table Flip, Death Counter). However, this reference focuses on the **base form** abilities, as per the scope. The base moveset alone is quite lethal – it has *very short average cooldowns (~16.25s on average)* [76] and straightforward high damage combos. Next, we cover cues and patterns an AI can use from these moves, and then dive into combo logic.

AI-Relevant Cues and Patterns for Moves

For a bot, each move in Saitama's arsenal provides **visual or audio cues** that can be detected, as well as UI indicators for the bot's own state. Here we enumerate cues useful for decision-making and timing:

- **Normal Punch (NP):** The cue is Saitama's dramatic wind-up animation (visible arm pull-back) accompanied by a swelling sound of air pressure. An enemy Saitama using NP is very obvious – the

bot should recognize this and **sidestep or get out of line**, since blocking won't help. If the bot itself uses NP, it should be aware of the long startup: ideally ensure the target is stunned or not facing us. The *impact* of NP is also a cue: a loud boom and shockwave effect on hit. If the bot hears/see this and its target suddenly goes flying, it knows NP landed (likely ending the combo there, as the enemy will be far away). If the shockwave hits (boom but the main punch missed), the target will be rolling away – the bot might then chase or switch target if that one is displaced.

- **Consecutive Punches (CP):** Visual cue: a rapid flurry of motion blur from Saitama's arms, dust kicking up. Audio: a machine-gun-like punching sound. If an opponent is caught in it, their character will be locked in a hit-react loop (flinching in place). For the AI, once it initiates CP, it knows the duration (~2s) it must *commit* – during this, it cannot do other actions, so it shouldn't attempt other inputs. However, the bot can prepare for follow-up: as CP ends, it should be ready to either continue with M1s or another move. If the bot sees an **enemy** using CP and it's not caught in it, this is a chance to punish (the enemy is stuck punching air). A quick dash behind or simply waiting out the 2 seconds then attacking is advisable. If *caught* in CP, the bot cannot do much except try an ultimate evasive if available (or hope the opponent mistimes the follow-up).

- **Shove:** It's very fast, so the cue is subtle – Saitama twists his body and elbows forward. There may be a whoosh and a distinct hit sound. Successful Shove hit produces a unique reaction: the target goes into a horizontal **rolling tumble** with a **yellow spin trail** effect. The bot can detect that state (a player tumbling head over heels) as a *"rolling"* state. This is important: a rolling opponent cannot be hit by normal high attacks unless you hit downward or wait till they stand. But Shove itself has the property to ragdoll a rolling opponent (as noted). Our AI uses a known **"Shove tech"**: if it lands a Shove and the target is sent rolling, the bot immediately holds M1 to attempt a follow-up hit right as the enemy exits the roll. Often this first M1 after a Shove will connect as a **"mini-uppercut"** (the target was low to the ground, so the M1 pops them slightly upward) [59]. Recognizing that visual (enemy popping up slightly from ground due to an M1) tells the bot its Shove tech succeeded, and it can go into further combo (e.g. Uppercut next). If the bot sees *its own Shove reflected a projectile* (the projectile flying back toward someone), that's a bonus – it might deal extra damage if it hits the opponent who threw it.

- **Uppercut:** Cue: Saitama crouches slightly then thrusts upward. Audio often a strong whoosh. The enemy will be launched in a distinctive arc (nearly straight up, then falling). The bot should visually track the launched target – turning the camera upward if needed – because it has a chance to **dash and catch** them. In code, the bot watches the target's **HumanoidRootPart Y-velocity**; when it sees the apex (velocity switching from up to down), it times a side/forward dash to where the target will land [77]. Visually, the bot can also use the enemy's shadow or falling animation as a cue to dash-catch. If the enemy uses an **evasive mid-air** (some characters might teleport out or dash in air), the bot will see the sudden change (e.g. the enemy flips out early). It should then adjust – perhaps turn toward where the enemy reappears and pursue. If an enemy Saitama uppercuts our bot, the bot could try to side dash **the instant it can** (on hit, you're stunned until you start falling; the moment that stun wears off, a side dash input can cancel the remaining fall). Recognizing that scenario (bot launched by uppercut) and using the evasive dash upon landing is key to avoid being combo-extended.

- **Wall Combo (Light Sneeze):** The cue for Saitama's wall combo is quite specific: if Saitama's normal 4th M1 knocks someone into a wall, an **"Impact!"** text or hit effect often appears. Then Saitama

dashes forward automatically and there is a unique *achoo* sound followed by a blast. Visually, there will be debris from the wall and the opponent will slump down at the base of the wall. For the AI, if it triggers this, it knows the sequence is automatic – it should not intervene until the sneeze animation completes (which is short). Right after, the enemy will be prone at the wall. The bot should press M1 immediately (as recommended by the game) to try to hit them as they get up [74]. If the bot *sees an enemy* perform a wall combo (perhaps to another player), it might consider third-partying at that moment since the attacker has a brief end-lag after the sneeze and the victim is downed (both are occupied). However, that's situational awareness that can be added.

- **Blocking and Critical Hits:** As mentioned, if the bot hears the **critical hit crack sound** [18], that means someone (could be bot or opponent) did a perfect block. For example, if the bot's M1 was perfect-blocked by the enemy, the bot should beware: the enemy's next punch will hurt a lot (3x damage) [17]. The UI doesn't directly tell this, but the audio/particle (red/black sparks) is the indicator. Our AI can incorporate this by not engaging in a trade right after the enemy got a crit – perhaps dash away or block the next hit. On the flip side, if the bot perfect-blocks an enemy's attack, it hears the sound and sees the indicator, and can confidently throw an M1 (it will be a powered **Critical**). A follow-up perfect block (two in a row) even triggers a **"Black Flash"** event (massive damage hit) [78], but this is rare and likely beyond initial AI scope.

- **HUD and Timers:** The bot continuously monitors its own cooldown timers via the hotbar (each move icon refills over time). We have functions like `slotReady(slot)` in the script to check if an ability is usable [27]. The bot will use this rather than relying on memory alone. For example, after using Shove (slot 3), it will not attempt Shove again until `slotReady(3)` returns true. For the enemy, the bot **cannot see their cooldowns** directly (that info isn't given to players), but it can infer approximate timers based on when the enemy last used a move. The bot internally "time-stamps" known enemy moves (especially the enemy's evasive dash, which we assume ~30s cooldown to be safe [79]) and uses that to decide if the enemy is likely able to use it again. This is part of perception-limited design: it **"mentally counts"** the cooldown since last seen use [80].

- **Position and State:** The bot pays attention to distance (studs) to target and states like rolling, ragdoll, blocking as discussed. For instance, if an opponent is **standing still** in a neutral pose, they might be typing or baiting – the bot might approach but cautiously. If the opponent is **holding block** (arms up), the bot may initiate a dash behind or use an unblockable like Shove or Uppercut. If the enemy is **attacking another player** and not targeting the bot, that's a window to attack them from the side/back. These are higher-level cues from behavior rather than specific moves, but important for decision.

In short, the AI should fuse all these cues – animations, effects, sounds, UI statuses – to make informed decisions in real time, much like an experienced player reacting to what they see and hear. We now move on to how the bot strings together moves into combos, and how it decides between **True Combos (TC)** and **Evasive-Counter Combos (ECC)** based on conditions.

Combo Decision Logic: True Combos vs. Evasive-Counter Combos

Saitama can perform devastating combos by chaining his moves in the right order. We categorize the bot's combo routes into **True Combos (TC)** – sequences that are generally inescapable *unless* the opponent uses their evasive – and **Evasive-Counter Combos (ECC)** – longer routes that assume the opponent has **no**

escape available, thereby maximizing damage. The bot's logic is to use TC routes when the enemy's evasive is likely up (to minimize risk and possibly force them to spend it) and to switch to an ECC route once the enemy **cannot** escape (e.g. their evasive was used or they are already caught in a true stun-lock).

True Combos (TC): These are relatively shorter combos that do significant damage (~40–70%) while leaving few gaps for the enemy to counter. A hallmark of a true combo is that from start to finish, the opponent is either in hitstun or ragdolled, with no free window to dash out (except a well-timed ragdoll cancel which we try to deny). An example basic true combo for Saitama is:

- **2 M1 → Consecutive Punches → 1 M1 → Normal Punch**, which reliably takes about **50%** of an opponent's health [81]. In this sequence, the first two M1s connect (target in stun), then CP keeps them stunned for 2s, then an immediate M1 hits as CP ends, and finally the Normal Punch is landed while they are still recovering – since NP is unblockable and hits like a truck, it will send them flying. Many players consider this "literally the most true combo in the game" for Saitama [82] – it's simple and effective. The wiki also notes Saitama has an "easy inescapable combo" around 59% [83], which is likely a variant of this.

Another true combo variant uses **Shove** as an opener:

- **1 M1 → Shove → (hold M1) → Side Dash → Consecutive Punches → 1 M1 → Normal Punch**, dealing about **61–68%** damage [84]. Let's break that down: one punch to start, then Shove (unblockable hit) puts them rolling. The bot holds M1 to smack them the moment Shove's roll ends – this produces a **"mini-uppercut"** (the target is popped up slightly by that M1) which flows right into a quick **Side Dash** forward (to close any gap) and immediately **Consecutive Punches**. The CP lands, then as it ends, one more M1, then the finishing Normal Punch. All of this is near seamless if done correctly. The side dash is in there to ensure after the mini-uppercut pop, we are still in range to continue. Because Shove prevents them from blocking and the follow-up hits are timed tightly, the opponent has no standard escape. Only a pre-used evasive (if they somehow spammed dash at the exact right millisecond between stun – very unlikely) or an external interruption would break it.

Other TC examples include: **2 M1 → Shove → Side Dash → 1 M1 → Uppercut** (about 35-40%) as a shorter combo [85], or **2 M1 → CP → Shove → (hold M1 to get 4th hit as mini-uppercut)** which yields ~42% [86] – this one uses both CP and Shove for moderate damage without needing NP or Uppercut finisher. The bot has a small library of these True Combos pre-scripted. It will choose one based on context (e.g. if Normal Punch is on cooldown, maybe use the Uppercut finisher combo instead; if both NP and Uppercut are ready and enemy is at say 40% HP, a shorter combo ending in Uppercut might suffice to then use NP on another enemy, etc.).

Evasive-Counter Combos (ECC): Once an enemy has burned their evasive (or if the bot is confident they cannot use it – for instance, they're in *true hitstun* from a downslam), Saitama can unleash his **most extended, high-damage combo strings**. These combos often approach 90–100% damage (one-shot territory) but they involve more steps and would be escapable if an evasive were available mid-way. Essentially, ECCs "overflow" the normal combo length, adding extra hits that normally would give the opponent a chance to escape *if* they could. The bot triggers ECC when `enemyHasEvasive == false` in its blackboard [87].

A prime example of an ECC (100% combo) is sometimes called the “Honey Badger” combo in the community [88], one version goes:

- **3 M1 → (jump) Downslam → Uppercut → Side Dash (catch) → 3 M1 → Shove → Front Dash → 3 M1 → (jump) Mini-uppercut → Normal Punch**, which basically uses everything in one sequence and will **KO a full health player** if done flawlessly [89] [90]. It’s very complex, but let’s simplify the logic: The bot does a full 3 or 4-hit M1 chain including a downslam (to ragdoll the opponent), then immediately Uppercuts the moment they bounce (because on wake from that ragdoll they couldn’t evade – true downslam setup [7]), launching them. Then it side-dashes to meet their fall, does 3 more M1s, then Shove to ragdoll again, then forward-dashes to keep close, then another 3 M1s, a mini-uppercut, and finally Normal Punch to finish. At multiple points this combo uses the fact the opponent has **no evasive**: after the first downslam they couldn’t escape the Uppercut; after Uppercut, they’re falling and can’t escape (because we assume no evasive or it was forced already); after the second ragdoll from Shove, again no escape, etc. This kind of combo is risky to attempt if you *aren’t* sure the enemy can’t dash out, because if they do escape at say the Uppercut, you’ve wasted all your cooldowns and might be punished. That’s why the bot only goes into this mode when it has confirmed the enemy is out of options (e.g. the enemy already used their side dash to escape something earlier, or the bot successfully did a true downslam which gives a guaranteed follow-up window).

A more practical ECC the bot might use is:

- **3 M1 (with a mini-uppercut on 4th) → Uppercut → Side-dash M1 → Consecutive Punches → 1 M1 → Shove → (hold M1) → Side Dash → Normal Punch**, which yields roughly **90–97% damage** [91]. This is an optimized sequence from our design: Start with 3 M1, and instead of downslamming on the 4th, do a **mini-uppercut** on the 4th hit (by jumping but striking before falling). This pops the enemy up slightly but doesn’t knock them away. Immediately follow with the **Uppercut skill** to send them higher. As they fall, side dash into an M1 catch, then **Consecutive Punches** to keep them in stun, then an M1, then **Shove** to ragdoll, then hold M1 to mini-uppercut them as they get up, then side dash and **Normal Punch** as the finisher. There are many steps, but each link is calculated: the mini-uppercut prevents the 4th hit from ending the combo prematurely, Uppercut adds big damage and airtime, CP after the catch ensures they stay helpless, Shove after that guarantees another ragdoll (which they can’t escape at that point), and NP seals the deal. If anything remained of their HP, one more tap would do it. The bot’s challenge is timing these precisely; fortunately as an AI it can execute frame-perfect sequences. The human limit is high here, but bots can hit it every time.

Another ECC example is a bit shorter:

- **2 M1 → Consecutive Punches → 1 M1 → (jump) Mini-uppercut → delay 0.2s → Normal Punch**, ~60% damage [92]. This one is useful if NP is ready and you’ve caught someone without evasive and maybe around half health – it will finish them off. The trick here is after CP, you do an M1 and a mini-uppercut (so basically 2 M1s in quick succession, where the second is done while jumping upward), then you wait a brief 0.2s (letting them drop just slightly to ground level) and fire the Normal Punch. The slight delay ensures the NP connects at the right height/distance for full effect. This combo is spacing-dependent (if you’re too far, NP could miss), but our bot would only use it when positioned well.

Route Selection Conditions: The AI's combo module will decide between TC and ECC based on a few factors: - **Enemy Evasive Status:** This is number one. If the enemy has *not* used an evasive in this engagement yet, the bot assumes they **can escape** a long combo. So it will choose a **True Combo** route that either forces the opponent to use the evasive or that minimizes the window for it. For instance, the bot might do a shorter 50% combo. Often, an opponent will panic and use their escape if they're getting hit by a combo that takes half their HP. If the bot sees the opponent **did not** or could not escape during that TC (e.g. they get fully hit and are now at half HP), it might mean the enemy had no chance or didn't have it available. At that point (especially if the bot still has some abilities off cooldown), it can attempt to continue pressure or go for an ECC next time. - **Enemy actually uses Evasive:** If during a combo the enemy successfully ragdoll-cancels (the bot will detect the enemy suddenly dashing out of a ragdoll or getting up quickly with a dash animation), then the combo is dropped. At that moment the bot will **switch to evasive-counter tactics**: often the best response is to **immediately block or dash** because many players who evasive behind you will try to hit you from behind. The bot can quickly turn or use a backdash to evade the counterattack (since it knows the enemy is likely to swing after escaping). Essentially, when the enemy uses an evasive, the bot needs to **abort the combo** and defend, or perform a counter-move if advantageous. After an enemy's evasive is spent, now the enemy has no escape for ~30 seconds (or whatever the dash cooldown is, typically side dash 2s but if they used their only one in a combo, practically there's a cooldown window). This is the time the bot will aim to perform an **ECC big combo** because the enemy can't get out of it. - **Bot's own cooldowns:** The availability of Saitama's moves dictates which combo routes are possible. For example, if **Normal Punch is on cooldown**, the bot won't choose a route that ends with NP; it might use Uppercut or a double downslam finish instead. If **Shove or CP is not ready**, that removes some combo extenders from the table. The AI will dynamically adjust. It may do a shorter combo or use a different extender (e.g. if CP is cooling down but Shove is available, use Shove tech; if vice versa, use CP mid-combo and maybe downslam instead of Shove). Saitama thankfully has short cooldowns, but managing them is key. The bot keeps track in its state of which moves are ready (as mentioned with `slotReady`). It will only initiate a combo that it can actually execute fully with the moves available. If few moves are available, the combo might just be a basic M1 chain and NP – not true or fancy, but sometimes necessary. - **Stage and Positioning:** The environment can influence route. If the fight has moved near a **wall**, the bot might alter its plan to incorporate the **Wall Combo**. For instance, if an enemy backs into a wall, the bot could opt for a quick M1 chain into wall sneeze, even if that wasn't the highest damage route, because it's guaranteed and gives armor during the wall combo [93]. Also, if the enemy is at **low health**, the bot might not need a full combo; it could do a simpler confirmed kill (why risk a long sequence if a short one will KO). For example, an enemy at 30% might just need one CP combo into NP. The bot will have logic to check target HP% and choose an efficient kill combo or even raw finisher if appropriate (ex: someone at 10% – just hit Normal Punch or even a few M1s). - **Multiple Opponents:** If the bot is in a free-for-all or being pressured by more than one enemy, long combos become dangerous (you can get hit by someone else while you are busy comboing one target). High-level players note Saitama lacks multi-target tools in base form and can be punished by groups [94]. The AI will incorporate an **"anti-gank" policy**: if surrounded or outnumbered, it won't do an extremely extended combo that leaves it stationary for 3+ seconds. It might instead do a shorter burst then **evasive dash out** to avoid getting hit by a third party. Or it may reposition such that any combo it does lines up enemies (so that a Normal Punch might hit one and shockwave the others). In 1v1, the bot can safely do full combos; in chaotic brawls, discretion is the better part of valor.

High-Level Strategies (Meta Insights): Here are a few strategic principles gleaned from the community and how we encode them for the bot: - **"Movement over combos"** – Top players often emphasize that staying mobile and avoiding enemy combos is more important than landing your own huge combo every time [43]. For the AI, this means it should not relentlessly chase with attacks at the cost of eating a counter-

combo. It will use its dashes and keep moving, only committing to a combo when a clear opening arises (e.g. enemy whiffed something or bot scored a guard break). This hit-and-run style keeps Saitama alive longer, especially given he has no extra armor in base kit. - **Baiting Evasives:** A common meta tactic is to do a partial combo to force the opponent's hand. For example, hit them with a quick combo that *could* be extended and see if they panic-escape. If they do, you *don't* continue (since they escaped), but now they're evasive-less – *then* you catch them with a full combo. The bot will emulate this. Early in a fight, it might deliberately drop a combo after say ~40% damage even if it could go further, just to see if the enemy uses their dash. If the enemy doesn't, perhaps they didn't have it or are saving it – the bot must judge. But if the enemy does use it, the bot immediately knows "no evasive now" and can go all-in. This aligns with the TC vs ECC logic. - **Tech Abuse:** The community has identified many "techs" (techniques) like true downslam, double downslam, shove-extenders, etc. We have integrated the key ones for Saitama. The **True Downslam** (3rd M1 immediate downslam) is used to ensure no escape at that point [7]. The **Double Downslam** tech (downslam → CP → downslam again) is actually possible: e.g. *3 M1 + downslam, then CP, then jump and downslam again* [95]. Saitama can do this – if timed well, the enemy might even end up in a weird "headstand" ragdoll that extends their vulnerability [96]. The bot could attempt double downslam combos for style or if it lacks other finishers. **Shove Tech** as described ensures a mini-uppercut after Shove, giving a guaranteed hit while the enemy is getting up [67]. **Uppercut Dash** tech (a known Garou/Hero Hunter tech applied to Saitama): dash immediately after a mini-uppercut to stay close so the enemy gets almost no knockback [97] – effectively what we do in our combos with side dash after mini-uppercut. These techs make combos tighter and more reliable, and the AI can execute them consistently (no human error). - **Counters and Defense:** The meta also involves using counters (certain characters have counter moves) and perfect blocks. While Saitama doesn't have a base counter (aside from the ultimate *Death Counter* in Serious Mode), the bot should be aware if fighting opponents who do. For example, if fighting a Hero Hunter (Garou) who has "Prey's Peril" counter, the bot must be careful throwing big moves or it could get insta-countered. This goes into opponent modeling, which is outside just Saitama's kit, but generally: the bot will incorporate a check, e.g., **don't use Normal Punch if the target is visibly in a counter stance** (some counters have an obvious animation). Instead, maybe bait it out with a fake or just wait. The user's question specifically asks for Saitama info, so suffice to say the bot will use block, dashes, and only spend its own evasive to **avoid getting caught in true combos** (e.g. if the bot gets downslammed and it has an evasive, it should use it immediately to avoid a big follow-up [98]). The bot's **Evasive Policy** is to only spend the evasive (side/back dash from ragdoll) to escape "true conversions" – i.e. when it knows the enemy landed something that guarantees more (like an uppercut about to be followed by a catch) [98]. It won't waste it on small hits that aren't leading to big damage.

- **Optimizing Damage vs Kill-secure:** Another high-level decision: sometimes it's worth doing a slightly lower-damage combo if it guarantees a kill or positioning. For example, if an enemy has 20% HP, doing a flashy 15-hit combo is overkill – a simple CP into NP might kill them with less risk. The bot evaluates the target's **HP percent** against its combo damage tables [99]. It knows approximate damage of each move and combo. It will choose a route that either kills or leaves the enemy in one-shot range with the least risk. Also, if using Normal Punch when an enemy is low, note that NP cannot *quite* outright kill if the target is above a certain threshold because NP leaves 1% if it doesn't slam them into something (due to how damage works, it can't one-shot from full, but from low it will) – the bot will be mindful if an enemy somehow survives an NP (very rare outside maybe Awakening interactions).

To wrap up, **Saitama's bot strategy** is to **engage with quick combos, force the opponent to blow their escape, then unleash a full combo to defeat them**, all while staying mobile and unpredictable. Community strategies like dashing behind blockers, combo extending with tech, and not falling for bait will

all be employed. By observing in-game cues (animations, sounds, UI) and following the routing rules above, the AI can perform at a top-tier level with “The Strongest Hero” – embodying the devastating, relentless offense Saitama is known for, but tempered with strategic movement and timing to ensure those hits land true [43] [40].

Sources:

- The Strongest Battlegrounds Wiki – *The Strongest Hero (Saitama) page* [100] [101] [55] [102], *Basic Combat mechanics* [103] [30] [5], *Combos and Techniques* [7] [67].
- Community discussions – e.g. Reddit combo tips [81] [89] and meta advice [43].
- Internal bot design notes (AI spec and script) – confirming move data and timings [104] [99] [91].

---

[1] [3] [4] [5] [6] [8] [9] [10] [12] [13] [14] [15] [16] [17] [18] [28] [29] [30] [31] [32] [33] [34] [39] [40] [70] [73] [78] [93] [103] Basic Combat | The Strongest Battlegrounds Rblx Wiki | Fandom
https://the-strongest-battlegrounds-rblx.fandom.com/wiki/Basic_Combat

[2] [11] [24] [25] [26] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54] [55] [56] [57] [58] [60] [61] [62] [63] [64] [65] [68] [69] [71] [72] [74] [75] [76] [83] [94] [100] [101] [102] The Strongest Hero | The Strongest Battlegrounds Rblx Wiki | Fandom
https://the-strongest-battlegrounds-rblx.fandom.com/wiki/The_Strongest_Hero

[7] [66] [95] [96] [97] Techniques | The Strongest Battlegrounds Rblx Wiki | Fandom
https://the-strongest-battlegrounds-rblx.fandom.com/wiki/Techniques

[19] [20] [21] [22] [23] [27] [36] [79] [104] aibot1.txt
file://file_000000008a2c71f5ae51e5338dcab92c

[35] [37] [38] dash stuff.txt
file://file_000000006f1071f78e11513b02ec58a7

[41] [42] [44] [77] [80] [84] [85] [86] [87] [91] [92] [98] [99] TSB_Bot_Spec_Saitama_v0_2.docx
file://file_0000000084d071f79722135decafca9e

[43] [81] [82] [89] Any effective combos of Genos or Saitama? : r/TheStrongestBattle
https://www.reddit.com/r/TheStrongestBattle/comments/1dr7gcn/any_effective_combos_of_genos_or_saitama/

[59] [67] [88] [90] Combos | The Strongest Battlegrounds Rblx Wiki | Fandom
https://the-strongest-battlegrounds-rblx.fandom.com/wiki/Combos